

Universidad de las Ciencias Informáticas



**Título: Sistema de gestión docente metodológica integrado
con la plataforma informativa SO3¹**

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

Autor(es): Arlety Campos León
Verónica Méndez Méndez

Tutores: Ing. Rosalina Ibarra González
Ing. Leonardo Darell Antúnez Naranjo

**Facultad 3
Junio, 2013**

¹ SO3: Plataforma de Servicios Integrados para la Facultad 3. Sistema que constituye el nombre de la plataforma informativa en desarrollo para la Facultad 3 que integrará todos los procesos que se desarrollan en la misma.

DECLARACIÓN DE AUTORÍA

Se declara que las tesis son las únicas autoras de este trabajo y autorizan a la Facultad 3 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Arlety Campos León

Verónica Méndez Méndez

AGRADECIMIENTOS

Agradecer primeramente ante todo a mi familia, principalmente a mi mamá, mi papá y mi hermano, por ser ellos, el motor impulsor de mi vida, guiándome y aconsejándome en cada paso dado. A mi tutora, que no solo le agradezco por guiarme en esta experiencia nueva para mí, sino por ser mi amiga, aquella persona que aunque no posee lazos sanguíneos de parentesco alguno, considero una hermana: Rosalina. A mi pareja, por soportar mis altas y bajas, a su familia de la cual soy parte: Raquel, Silvio, Yoelquis y su amigo Yaiquel. A mis viejos amigos de la antigua facultad 4, en mi primer y segundo año, y a los nuevos de la actual facultad 3: mis compañeros de aula. A todas mis amistades de Banes, que por la lejanía no pudieron compartir este momento conmigo, y los que sí están presentes aquí hoy. A mis compañeras de cuarto por soportarme con tanto cariño: Yadelis, Diana, Dania, Yane... A mi cotutor y mi compañera de tesis. Al tribunal, por apoyarnos y confiar en nosotras. A los nuevos amigos del laboratorio, principalmente a Reinier, porque sin él, el laboratorio nunca abriría por la noche (noches de duro trabajo), ni los fines de semana. Aquellos amigos que no puedo dejar de mencionar por ser los que me ayudaban, a si fuese unos minutos, cuando yo me enredaba y me sentía perdida en la programación: Eidy, Luis Manuel y Robin. Agradezco a todos los presentes por estar aquí y confiar en nosotras.

Dedicatoria

Dedico este trabajo a mi mamá, a mi papá y a mi hermano, por desear y hacer cumplir mi sueño de convertirme en una ingeniera al igual que ellos, besos, los amo mucho.

Verónica

AGRADECIMIENTOS

Le doy gracias al señor por permitirme levantarme cada vez que he tropezado durante estos 5 años. A mi mamá, que tanto ha sufrido y pedido por mí, a papá por hacerme testaruda una cualidad que a pesar de ser mala, me ha llevado a este momento. A ambos por apoyarme, quererme y tratar de guiarme estos 22 años. A mis tíos Dignora, Francisca y Henry por escucharme, aconsejarme y malcriarme. A mis hermanas Geidy y Yisel por complacerme y quitarme trabajo en muchas ocasiones, además de darme fuerzas para continuar con las cosas. Al resto de la familia que de una manera u otra aportaron a que fuera posible continuar aquí. A mis tutores Rosalina y Leonardo por ayudarme en todos los momentos, soportar mi carácter y crisis de estrés. A Yorlen y Eddie por fajarse conmigo y aun así después seguir ayudando en lo que necesitara. A mis amistades de Cienfuegos Luis Manuel, Alberto y a todos los que conocí aquí. A los amigos que hice en estos 5 años Rosalina, Arianna, Katy, Verónica, Polanco, Abel, Eidy, Ariel, Pastor, José Luis y Darien con los cuales compartí, aprendí, trabajamos, fiestamos y sufrimos juntos. Todos son muy importantes para mí, los quiero y espero no me olviden.

Dedicatoria

Dedico este trabajo a mis padres que sin darse cuenta me motivaron a cumplir algunos de sus sueños y de querer hacerlos sentir orgullosos.

Arlety

RESUMEN

La Universidad de las Ciencias Informáticas se ha caracterizado por gestionar grandes volúmenes de información, que en su mayoría debe ser accesible por las autoridades competentes.

En la Facultad 3, de dicha universidad, tienen lugar los procesos Plan de trabajo metodológico, Guardia docente y Control a clases los cuales son ejemplos claros del manejo de gran cantidad de información, generación de estadísticas y toma de decisiones. De acuerdo a las actividades realizadas para lograr la finalidad de cada uno, se evidencia que la información está descentralizada y se gestiona manualmente, conllevando en cada una, aumento de la complejidad, afectándose por errores, omisiones, retrasos y desactualización. Además de que el acceso simultáneo a la información se ve comprometido porque se realiza directamente con la secretaria, lo cual conduce a demoras en su respuesta por diversos problemas.

El presente trabajo de diploma tiene como objetivo dar solución a las diferentes deficiencias concerniente a los procesos antes mencionados, permitiendo la centralización, así como una adecuada organización y una eficiente gestión de la información. Además de proveer una retroalimentación de datos entre los involucrados, la generación de estadísticas, una mayor disponibilidad y actualización de la información gestionada, por lo cual surge la necesidad de comenzar un nuevo proceso de desarrollo, conllevando a la obtención del sistema MOIRA.

Como resultado final de la presente investigación se obtiene una aplicación web capaz de centralizar y gestionar la información de los procesos docentes metodológicos adscritos a la Facultad 3 de la Universidad de las Ciencias Informáticas.

PALABRAS CLAVE

Centralización, control a clases, docente metodológico, Facultad 3, gestión, guardia docente, información, plan de trabajo metodológico.

CONTENIDO

AGRADECIMIENTOS.....	I
AGRADECIMIENTOS.....	II
RESUMEN.....	III
INTRODUCCIÓN.....	1
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.....	7
1.1 Introducción	7
1.2 Análisis terminológico-conceptual	7
1.2.1 Definición de proceso.....	7
1.3 Soluciones informáticas de gestión docente metodológica	9
1.3.1 Sistemas de gestión docente metodológica internacionales	9
1.3.2 Sistemas de gestión docente metodológica nacionales.....	12
1.4 Selección de las tecnologías, herramientas y metodologías	15
1.4.1 Metodologías de Desarrollo de Software.....	16
1.4.2 Herramientas de Ingeniería de Software Asistida por Computadora (Visual Paradigm 8)	18
1.4.3 Herramienta para el desarrollo de prototipos de IU (Axure).....	19
1.4.4 Notación de Modelado de Procesos de Negocio (BPMN).....	20
1.4.5 Lenguaje Unificado de Construcción de Modelos (UML)	20
1.4.6 Lenguajes de programación (PHP 5.3).....	21
1.4.7 Twig	22
1.4.8 Marco de trabajo	23
1.4.9 Bibliotecas (Jquery 1.8).....	24
1.4.10 Sistema Gestor de Bases de Datos (PostgreSQL 9.1)	25
1.4.11 Entorno de Desarrollo Integrado (Netbeans 7.2)	25
CONCLUSIONES	26
CAPÍTULO 2: CONSTRUCCIÓN DE LA PROPUESTA	27
2.1 Introducción.....	27
2.2 Descripción del flujo actual de los procesos.....	27
2.2.1 Proceso de gestión Guardia Docente	27

2.2.2	Proceso de gestión de los Controles a clases.....	27
2.2.3	Proceso de gestión Plan de trabajo metodológico	28
2.3	Mapa conceptual	29
2.4	Modelo de proceso del negocio Plan de trabajo metodológico	30
2.5	Descripción del proceso del negocio Administrar Plan de trabajo metodológico	34
2.6	Técnicas de recopilación de requisitos	36
2.6.1	Requisitos funcionales	37
2.6.2	Prototipo de Interfaz de Usuario	39
2.7	Modelo Entidad Relación (MER)	41
2.8	Patrones.....	42
2.8.1	Patrón arquitectónico Modelo Vista Controlador (MVC)	42
2.8.2	Otros patrones de Software	46
2.9	Diagrama de Clases del Diseño.....	50
	CONCLUSIONES	51
	CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS.....	52
3.1	Introducción	52
3.2	Estándares de codificación	52
3.3	Validación de los artefactos	52
3.3.1	Métricas de software.....	53
3.4	Pruebas	58
3.4.1	Pruebas del nivel de Unidad	59
3.4.2	Pruebas del nivel de Sistema	64
	CONCLUSIONES	66
	CONCLUSIONES.....	67
	RECOMENDACIONES.....	68
	BIBLIOGRAFÍA.....	69

INTRODUCCIÓN

La evolución de la humanidad contempla el desarrollo de la ciencia y la tecnología como base para su adaptación y supervivencia. Por lo que resulta imprescindible la transformación del conocimiento adquirido, en nuevas experiencias que propicien el avance de la sociedad en sus diferentes manifestaciones. Dicho desarrollo se evidencia en diversos entornos contribuyendo sustancialmente al perfeccionamiento de los medios y métodos tradicionales empleados por el hombre.

De ahí que las Tecnologías de la Informática y las Comunicaciones (TIC) tienen un papel importante en la traducción de problemas de la vida real, en soluciones objetivas que enriquezcan el razonamiento lógico y abstracto de todos los implicados. Dichas soluciones surgen del análisis crítico y profundo de diversas problemáticas que tienen como trasfondo alguna insuficiencia en la ejecución de un proceso o en la gestión realizada por el hombre. Dichas premisas, son sustentadas con el hecho de que algunos procesos son realmente complejos cuando involucran grandes cantidades de información estadística, actualización de datos sensibles para facilitar la toma de decisiones y una amplia retroalimentación entre diferentes áreas, donde cualquier retraso o imprecisión puede afectar seriamente a la institución o negocio en cuestión.

Ante tales inconvenientes, muchas empresas de diversa índole, así como universidades, centros de investigación e incluso administraciones de gobierno han implementado planes de desarrollo tecnológico que progresivamente vayan estrechando los canales de comunicación entre todas las partes implicadas. De modo que la calidad y la eficiencia en la ejecución de las diversas tareas proyectadas, sean una meta objetiva a medida que se persigue mantener la competitividad en el mercado internacional. Estos objetivos son afines con la estrategia de desarrollo del gobierno cubano, que persigue la informatización de la sociedad no solo para alcanzar todos los beneficios antes expuestos, sino para contribuir con la superación del pueblo y la recuperación económica ante el genocida bloqueo y las diversas agresiones del imperialismo yanqui.

De los esfuerzos que realiza el gobierno y el pueblo de Cuba, surge como resultado la Universidad de las Ciencias Informáticas (UCI) creada en el 2002. Emerge como punto de partida para la producción de software de uso nacional y para la exportación, así como para la formación de profesionales altamente calificados y comprometidos con la Revolución. En

el cumplimiento de este último compromiso, las diferentes facultades de la UCI gestionan grandes volúmenes de información que en su mayoría debe ser accesible por las autoridades competentes teniendo en cuenta todos los requisitos de seguridad necesarios.

La presente investigación, siguiendo el principio de la informatización y el desarrollo tecnológico del país, pretende proponer una solución a la siguiente problemática identificada en el área de la Facultad 3:

A pesar de que existen varias aplicaciones informáticas para el procesamiento de datos de diversa índole, aun se tienen deficiencias para proveer una retroalimentación de datos, la generación de estadísticas, una mayor disponibilidad de la información y actualización de la misma. Tal es el caso de los procesos Plan de trabajo metodológico, Guardia docente y Control a clases, para los cuales toda la gestión de la información es manual y al proveer un gran número de cifras y referencias estadísticas, resulta muy engorroso, complejo y en ocasiones inexacto el tratar de derivar conclusiones a partir de los registros de cada uno de ellos. Lo que para el tratamiento y resolución de problemas como violaciones en la ejecución del proceso docente educativo, en el cumplimiento de los reglamentos docentes y resoluciones ministeriales, dificulta la disponibilidad de información para respaldar la toma de decisiones en los distintos niveles de la administración.

Por ejemplo, en el caso del proceso relacionado con el control a clases, es perceptible la dificultad para la recolección de todos los datos en forma digital limitándoles a los jefes de departamento, al vicedecano de formación o al decano conocer reportes estadísticos sobre las evaluaciones del claustro, las evaluaciones promedio del mismo en una asignatura, la calidad de las clases, así como trazar planes con mayor alcance para el tratamiento de problemas concretos que puedan presentar los profesores o el programa de alguna asignatura. Al estar la plantilla del control en formato duro, dificulta aun más la recolección de datos específicos sobre un indicador, deficiencias encontradas que puedan ser comunes para varios casos o simplemente poder acceder al cúmulo total de los principales logros de un profesor a lo largo de un semestre o curso escolar, sin tener que revisar todos y cada uno de los controles a clases realizados.

En adición la información del Plan de trabajo metodológico de cada departamento está fuera del alcance del decano o el vicedecano, lo cual les dificulta comprobar la existencia de un conjunto de actividades que tributen al cumplimiento de las orientaciones de la Universidad y

de la Facultad. Así mismo, para reflejar la información de la guardia docente, debe emplearse gran cantidad de papel y otros materiales de oficina, lo que atenta contra el ahorro de recursos y la organización de los datos.

Desde el punto de vista de la capacidad humana, cuando todos los reportes y la gestión estadística de la información dependen de una sola persona el proceso puede verse comprometido por errores, omisiones, retrasos y desactualización. Además de que el acceso simultáneo a la información por las distintas autoridades inherentes al negocio se ve comprometido porque se realiza directamente con la secretaria, lo cual conlleva a demoras en su respuesta por cuestiones logísticas u organizacionales, aunque algunas veces los problemas personales o de otra índole pueden afectar su disponibilidad. Es importante destacar que como muchos de estos datos se manejan vía correo, si falla la conexión puede dificultarse el envío, transmisión, recepción y actualización de la información.

Evidentemente, los problemas están relacionados con la gestión realizada por el personal y no a incongruencias en los procesos como tal, por lo que con basamento en la problemática antes expuesta se plantea el siguiente **problema a resolver**: la forma en la que se gestionan los procesos docentes metodológicos de la Facultad 3 no favorece actualmente una mayor disponibilidad, accesibilidad y actualización de los datos entre los miembros del claustro. Por tanto el **objeto de estudio** de esta investigación se centra en la gestión de los procesos docentes metodológicos y el **campo de acción** está enmarcado en los procesos Guardia docente, Control a clases y Plan de trabajo metodológico.

Como **objetivo general** se pretende: informatizar los procesos de Guardia docente, Control a clases y Plan de trabajo metodológico para contribuir con la disponibilidad, accesibilidad y actualización de los datos inherentes a los mismos.

A partir de la investigación anterior, se plantea la siguiente **Idea a defender**: la informatización de la gestión de los datos correspondientes a la planificación, seguimiento y control de la actividad docente-metodológica de la Facultad 3 contribuirá con el incremento de la disponibilidad, accesibilidad y actualización de los datos entre el claustro.

Para dar cumplimiento al objetivo general de esta investigación se proponen los siguientes **objetivos específicos**:

- Establecer el marco conceptual de referencia.
- Modelar la solución.
- Diseñar la solución.
- Implementar la solución.
- Validar los resultados.

Para llevar a cabo este trabajo y cumplir con los objetivos específicos propuestos se concibieron las siguientes tareas de investigación:

- Realización de búsquedas y entrevistas preliminares sobre el estado del arte de la investigación para determinar pertinencia.
- Elaboración del estado del arte de la investigación.
- Realización del marco teórico de la investigación.
- Identificación de los procesos de negocio.
- Modelación de los procesos de negocio.
- Construcción del mapa conceptual.
- Descripción de los procesos de negocio.
- Identificación de los requisitos del negocio.
- Descripción de los requisitos del negocio.
- Identificación de las entidades críticas para el diseño de la Base de Datos.
- Diseño del Modelo Entidad-Relación de la Base de Datos.
- Diseño de los diagramas de clases del sistema.
- Definición del estándar de codificación.
- Implementación del sistema.
- Definición de tipos de pruebas para validar software.
- Diseño de casos de prueba para validar el sistema.
- Aplicación de casos de pruebas para validar el sistema.
- Corrección de las No Conformidades de la validación.

Se realizó una estrategia descriptiva denotándose las características por las cuales se evidencia una existencia clara de la problemática a investigar y haciéndose énfasis en los procesos esenciales. De ahí que los métodos y técnicas de investigación que se emplearon fueron los siguientes:

Métodos teóricos:

- Analítico-Sintético: a través de este método se definieron los principales elementos que caracterizan a los sistemas de gestión, así como las herramientas más utilizadas en el desarrollo de éstos, tomando como punto de partida los aspectos relacionados con la gestión de la información docente metodológica.
- Modelación: este método se llevó a cabo mediante el análisis y diseño del sistema de gestión docente metodológica a través de las diferentes herramientas que se seleccionaron, así como de la implementación para explicar las principales estructuras del sistema desarrollado.
- Histórico - Lógico: con la utilización de este método se realizó el análisis de la trayectoria de la metodología de desarrollo de software y herramientas que se usaron en el proyecto. Además del estudio de la gestión de los procesos comprendidos en el área de la Facultad 3.

Métodos empíricos:

- Entrevista: con el objetivo de lograr un mejor entendimiento del negocio se realizaron entrevistas y encuentros con el cliente, de manera que a través de listas de preguntas se identificó, de acuerdo a su nivel de importancia, toda la información a ser reflejada en el desarrollo de esta investigación.
- Medición: mediante este método se determinó un procedimiento con el objetivo de controlar los resultados de la evaluación, para la medición de la propiedad o calidad de cierto elemento. De manera que al concluir el desarrollo de la investigación se tuvieran todos los elementos para validar los resultados que se obtuvieron en función de determinar si se solucionó el problema planteado.

En aras de lograr un mayor desglose durante la evolución de este trabajo, se estructuró el mismo en 3 capítulos, los cuales se resumen a continuación:

Capítulo 1: Fundamentación Teórica

Se abordan los conceptos y tendencias principales referentes a los sistemas de gestión vinculados a lo docente metodológico en el ámbito internacional y nacional. Se realiza un

estudio de las posibles técnicas y herramientas a emplear para desarrollar la propuesta de este trabajo, así como la sustentación de todas las selecciones realizadas.

Capítulo 2: Construcción de la propuesta

Se obtiene la modelación de los procesos del negocio y sus descripciones, el mapa conceptual, la lista de requisitos funcionales y no funcionales con sus respectivas descripciones. Además se construyen los prototipos iniciales de interfaz de usuario y se diseñan los diagramas de clases del sistema.

Capítulo 3: Implementación y Prueba

A partir de la propuesta del capítulo 2 de la investigación, se procede al desarrollo y validación del sistema a través de pruebas de software con el objetivo de detectar errores en la implementación de los requisitos.

Al finalizar el registro de la investigación en el documento, se encontrarán las conclusiones, recomendaciones, las referencias bibliográficas, anexos y el glosario de términos.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se abordará una aproximación a los temas relacionados con los sistemas de gestión de manera general, dándole un enfoque principal a los relacionados con los procesos que se desean informatizar. Llegando así, a desarrollar las aristas más significativas de cada uno de ellos. Se pretende además exponer el contexto nacional e internacional de dichos sistemas realizándose en estudio crítico y valorativo partiendo de sus características más significativas; en pos de poder corregir las deficiencias existentes e incluir aquellas funcionalidades que aún no son gestionadas como el cliente desea. Además se describirán un conjunto de herramientas, tecnologías y modos de actuación que responden al contexto en el que fuera desarrollada la primera aplicación de SO3, a la cual la propuesta de este trabajo deberá integrarse.

1.2 Análisis terminológico-conceptual

El análisis terminológico-conceptual tiene gran connotación en los trabajos investigativos, debido a que tienen generalmente numerosas definiciones y acepciones, por lo cual pueden ser usados de diferentes modos. Además, realizar este análisis evita la confusión de términos entre diferentes conceptos.

1.2.1 Definición de proceso

Según Luis Bernardo Osorio, un proceso no es más que: *“Cualquier actividad o grupo de actividades que emplee un insumo, le agregue valor a este y suministre un producto a un cliente interno o externo.”* Definiendo también que: *“... Un proceso de la empresa consiste en un grupo de tareas lógicamente relacionadas que emplean los recursos de la organización para dar resultados definidos en apoyo de los objetivos de la organización”* (Osorio, 2008).

Enmarcándose en los procesos que se desarrollan en el vicedecanato de la Facultad 3, se puede apreciar una serie de pasos que se traducen por etapas o fases, haciendo uso de diversos recursos y son realizables con la finalidad de lograr un resultado planificado.

1.2.1.1 Plan de trabajo metodológico

El plan de trabajo es la definición de qué se va a hacer y por qué, obedeciendo siempre a objetivos planificados previamente para reducir el riesgo del negocio en determinadas situaciones y anticipando las consecuencias del mismo. Además de permitir organizar en todo momento los elementos involucrados, en aras de elevar el éxito organizacional, efectivo de las acciones desarrolladas y alcanzar dichos objetivos planificados.

El Plan de trabajo metodológico será confeccionado a partir de los problemas detectados durante el proceso docente, los cuales serán el punto de partida para constituir los objetivos a alcanzar. Luego, dichos objetivos se concretan en acciones específicas a desarrollar para cada curso académico, utilizando las formas y tipos que se establecen en el Reglamento Docente y Metodológico del Ministerio de Educación Superior de Cuba. Dicho plan deberá ser confeccionado por cada uno de los colectivos y niveles de dirección al inicio de cada curso y podrán adecuarse para cada período escolar. A nivel de Facultad todos los documentos deberán ser conservados en el departamento docente para avalar el trabajo metodológico después de ser aprobado por el decano, a su vez se conservará en las facultades el Plan de trabajo metodológico de la Facultad aprobado por la dirección del centro de educación superior (Ministro de Educación Superior, 2007).

1.2.1.2 Control a Clases

Según el Reglamento Docente y Metodológico, Resolución No. 210/07, en su artículo 90 se refiere al control del proceso docente educativo como que (Ministro de Educación Superior, 2007): *“... es parte del contenido del trabajo metodológico y el medio fundamental para conocer la calidad de dicho proceso, evaluar sus resultados y dirigirlo hacia el cumplimiento de sus objetivos...”*. Donde se establece además que el control a clases del proceso docente educativo comprende el control a la planificación y organización, el control a la calidad de la ejecución, el control a la actividad docente, el cumplimiento de los programas de estudio, el aseguramiento material y humano de las actividades docentes y el control a la preparación adquirida por los docentes. Para ello, los controladores deben tener una categoría docente igual o mayor que la del controlado y la cantidad de controles dependerá de la maestría pedagógica y del dominio del contenido que tenga el docente de la asignatura que imparte (Castillo González, y otros, 2009-2010).

1.2.1.3 Guardia Docente

La Guardia Docente, se realiza por mediación de un profesor/controlador que recorre todos los locales donde se debe estar impartiendo clases para controlar de manera general el cumplimiento del reglamento que establece las pautas de disciplina a ser cumplidas por los profesores y estudiantes. Por lo que constituye un mecanismo de control sobre el proceso docente. Los datos más importantes de cada registro están relacionados con la correspondencia según el horario de los grupos, el aula, la asignatura y el profesor que debe estar impartiendo la clase en el momento de la revisión. Por otro lado también se registran las incidencias ocurridas, aportando la mayor cantidad de información sobre el hecho en cuestión para el posterior análisis (Batista, 2013).

En lo concerniente a la gestión docente metodológica, deben definirse las pautas sobre las cuales es posible justificar la informatización de los procesos antes mencionados. Por tanto, el propósito de planificar, organizar, dirigir, motivar, controlar y tomar decisiones durante todas las acciones que tributan a la educación y que se realizan basándose en los conocimientos más consolidados de la didáctica general y especial, y en la preparación y experiencia acumuladas por los profesores genera información complementaria que resulta vital para cualquier universidad manejarla con rapidez y actualización, con el fin de mejorar el proceso de enseñanza aprendizaje. En aras de establecer un punto de partida con las soluciones de software dedicadas a la gestión de dichas actividades, se agregan los siguientes epígrafes.

1.3 Soluciones informáticas de gestión docente metodológica

En el momento de gestionar un negocio la dirección de una empresa requiere de un conjunto de tareas realizables para desarrollar un proceso o lograr un producto. Necesitando en todo momento de recursos, una dirección y un liderazgo para obtener el logro de los objetivos. La gestión docente metodológica no está exenta de estas necesidades, requiriendo en todo momento de docentes, discentes y el objeto de conocimiento. Es por ello que se hace necesario el entendimiento de estos sistemas, los mejores escenarios de los mismos y el aporte al negocio.

1.3.1 Sistemas de gestión docente metodológica internacionales

Se evidencia la existencia y uso temprano de los sistemas de gestión en un negocio u empresa a nivel internacional debido al desarrollo existente en otros países, que conllevó al surgimiento de los mismos antes de los existentes a nivel nacional. Su estudio puede acarrear el aprendizaje de sus experiencias en la realización, para luego aportar a dicha investigación un enfoque diferente. Por ello a continuación se estudiarán algunos de los sistemas relacionados con el tema de la investigación.

1.3.1.1 Gestión de centros de enseñanza (KGW32Plus)

Es un programa informático que gestiona los centros de enseñanza. Brinda algunas funcionalidades docentes como son: realizar evaluaciones, llevar el control de faltas de alumnos, además del registro de entradas/salidas de documentos del centro y permite exportar a diversos formatos (xls, pdf, html, entre otros). En la actualidad presenta varias versiones las cuales fueron creadas por la empresa ADOSSIS, S.A². Hace uso de varias bases de datos, aporta mayores posibilidades de configuración y adaptación a las características del centro docente, genera mayor número de informes y estadísticas preestablecidas.

Características (ADOSSIS S.A., 1985-2012):

Es funcional para los entornos de Windows 95, 98,98 SE, XP y NT (solamente para Windows de 32 bits) y se encuentra en la versión 5. Permite ganar tiempo en velocidad de manejo y facilidades de uso, además de presentar un lenguaje orientado a objetos.

1.3.1.2 Colegium.com

Colegium es una empresa chilena que nació en el año 2000 la cual desarrolla e implementa aplicaciones informáticas especializadas en gestión escolar. Dicha empresa reduce el tiempo de

² Empresa que fue constituida en Valladolid en Noviembre de 1985, desde su fundación, ha dirigido su actividad hacia áreas determinadas dentro de los sectores de informática y electrónica, basados a lo largo de estos años, en el sector de la enseñanza. Así se han creado diversas aplicaciones ampliamente distribuidas y orientadas a este sector tales como, Kronowin Generador de Horarios, Kgwin32 Gestión de Centros Escolares, Kgrupos Generador de Grupos de Alumnos, Kodex Generador/Organizador de Exámenes, Kbiblos Gestión de Bibliotecas. Esta última se encuentra avalada por las más de 8.000 licencias instaladas en la actualidad.

procesos administrativos como: calcular promedios, administrar bibliotecas, enviar comunicaciones y circulares y transcribir calificaciones al libro de clases. Aporta seguridad en el manejo de la información, haciendo uso de tecnología mixta y maneja información académica, financiera, de admisión, transporte, bibliotecas existentes, sobre profesores, alumnos y padres. Algunos de sus resultados se evidencian en Argentina, Chile, Colombia, México, Venezuela, Brasil y Perú. En estas aplicaciones interactúan los alumnos, directores, profesores, funcionarios, administradores y padres de familia (Colegium.com, 2012).

Características:

Mejora el trabajo de profesores y administradores, manteniendo una mejor supervisión de la escuela por parte del director y mantiene los sistemas actualizados en dependencia del ministerio de educación de cada país. Permite configuración de privilegios para cada perfil por lo cual cada alumno y padre puede acceder bajo su propio perfil brindando una comunicación más eficiente, rápida y directa. Es un software con más de 20 años de experiencia que ha integrado las mejores prácticas de gestión escolar de Latinoamérica.

1.3.1.3 Sistema Integral de Gestión Académica (CAMS)³

Three Rivers Systems Inc.⁴ es un proveedor líder de soluciones de software para la administración superior de educación el cual desarrolló el Sistema Integral de Gestión Académica (CAMS) haciendo uso de tecnologías de Microsoft. Éste es altamente escalable, rápido de implementar y fácil de usar, manteniendo siempre la información más reciente y precisa, tratando de satisfacer las necesidades de administradores, personal, profesores y estudiantes. Entre las universidades que usan este sistema se encuentra la William Jessup University (WJU)⁵ de California que lo utiliza para gestionar el ciclo de vida estudiantil. Es totalmente integrable, y sigue los estándares establecidos por los sistemas de planificación de recursos empresariales (ERP)⁶, se adaptó exclusivamente para la educación superior. Dentro de las funciones que maneja se pueden evidenciar portales, registros, facturación, ayuda

³ Del inglés: Comprehensive Academic Management System.

⁴ Con sede en St. Louis es el desarrollador y proveedor de servicios para CAMS, la principal solución global de Gestión Académica de la Educación Superior.

⁵ Por su nombre en inglés William Jessup University.

⁶ Del inglés: Enterprise Resource Planning.

financiera, gestión de cursos, recaudación de fondos y Gestión de Relaciones con el Cliente (CRM)⁷. Además ofrece servicios de implementación, capacitación, gestión de proyectos y el apoyo a sus clientes. Consolida múltiples sistemas independientes, incluidos las admisiones, la información del estudiante, relaciones con ex alumnos y gestión de documentos (Three Rivers Systems, 2010).

Características:

Es una solución integrada, que presenta una profunda inteligencia de negocios, mayor accesibilidad y mejor informatización y comunicación. Brinda un rápido proceso de implementación y de transición al nuevo sistema, con una total integración con la base de datos MySQL. Presenta gran funcionalidad y escalabilidad brindándole una buena gestión de sus procesos y elimina la necesidad de las instituciones académicas y de los usuarios finales tanto para aprender interfaz como para gestionar un conjunto de sistemas que a menudo son incompatibles.

De manera general, se puede concluir que estos sistemas no satisfacen las necesidades del cliente. Además de que para su confección emplean tecnologías privativas y los procesos que gestionan no son totalmente compatibles con el objeto de estudio de esta investigación. Las herramientas usadas no son compatibles con el sistema SO3 ni con la plataforma tecnológica definida en la **tabla 1**, mostrada más adelante. Es preciso entonces, enfocar el estudio hacia los sistemas nacionales.

1.3.2 Sistemas de gestión docente metodológica nacionales

La nación cubana se ha enfocado en la gestión de los negocios en aras de administrar mejor sus recursos, haciendo uso solamente de los imprescindibles y aportando una dirección y liderazgo provistos de facilidades para controlar y guiar en aras de alcanzar los objetivos planificados. Por este motivo existen actualmente varios sistemas, dentro de los cuales se podrán encontrar algunos relacionados al contexto docente metodológico, tal es el caso de los mencionados a continuación.

1.3.2.1 Sistema para la gestión académica (Gestacad)

⁷ Del inglés: Client Relationship Management.

Sistema para la gestión académica existente en la Universidad de Matanzas “Camilo Cienfuegos”, específicamente en la Facultad de informática. Permite actualizar y mantener la información sobre estudiantes y profesores de una universidad a través de un sistema cliente que tiene acceso a una base de datos existente en un servidor. Dentro de sus funcionalidades se podrán encontrar la actualización de notas, búsqueda de un alumno y las notas del mismo. Además de asignar grupos a profesores, estudiantes a grupos y la actualización de los datos de estudiantes. También se puede obtener la carga docente de un profesor y realizar consultas vía WEB (dentro de las cuales se pueden notar las de datos personales de los estudiantes, cantidades por sexo, vías de ingreso). Permite brindar informaciones sobre los resultados docentes (de forma individual, por asignatura o por alumno) que se pueden mostrar como tablas que se calculan directamente a partir de los datos existentes.

Características (Dr. Telot González, y otros, 2001):

Permite obtener información desde distintos lugares de la red universitaria usando el formato web e incluso a través de Internet. En dicho software se informatiza el procesamiento de informaciones docentes de pregrado, postgrado, ciencia y técnica entre otras, facilitando así la obtención de información con características como mayor eficiencia y rapidez para lograr tomar decisiones importantes en una institución universitaria. Usa para la gestión de base de datos a Interbase y Firebird, usando la tecnología ActiveX Data Object (ADO)⁸ para la conexión a la base de datos remota, a la cual diferentes usuarios tienen acceso. El servidor de base de datos es a su vez servidor WEB y los resultados se obtienen a través de Páginas Activas en el Servidor (ASP)⁹ que tienen acceso a los datos del servidor de bases de datos. El sistema ha sido empleado con estudiantes del centro durante un curso, permitiendo obtener información del funcionamiento del sistema por las diferentes áreas implicadas.

1.3.2.2 Sistema de gestión para la planificación docente en la Facultad 5 (SGPD)

En la Facultad 5 se desarrolló un sistema de gestión con la finalidad de planificar y controlar procesos docente educativos. Dentro del cual se puede notar el cuidado de exámenes y la planificación de la guardia docente (teniendo en cuenta la planificación de locales, horarios,

⁸ Microsoft ActiveX Data Objects (ADO) permite a las aplicaciones clientes acceder y manipular la información de una variedad de fuentes en la Base de Datos.

⁹ Del inglés: Active Server Page.

distribución de profesores con sus respectivas afectaciones como las preparaciones metodológicas y horarios de superación).

Características (Alvarez Hernández , y otros, 2009):

El sistema hace uso como gestor de bases de datos de MySQL y como lenguaje de programación: PHP del lado del servidor y JavaScript del lado del cliente. Permite la planificación eficiente del cuidado de exámenes y la guardia docente. Presenta rapidez y organización en su ejecución, aportando resultados favorables en cuanto a su rendimiento y aceptación del cliente final. A pesar de tener las características ya mencionadas al llegar a su culminación no se puso en práctica y no alcanza otras funcionalidades de los procesos comprendidos en el área de formación de la Facultad 5. Además de no integrarse con el sistema de generación de horario que fue desarrollado en otro trabajo de diploma.

1.3.2.3 Sistema de gestión de información de la Facultad 8. Módulo para la gestión de la información

La gestión de información docente de la Facultad 8 de la UCI fue informatizada en un sistema que comprende el control de: las visitas a clase, los alumnos ayudantes, los departamentos docentes, las evaluaciones del personal docente, los planes de trabajo del personal docente y la aplicación de las resoluciones decanales. Dentro de las facilidades que brinda, se encuentra obtener un listado de los docentes que imparten clases en un departamento determinado y obtener quienes son los jefes de los distintos departamentos. Ofrece la información de las evaluaciones que se les realizan a los profesores mensualmente y a los alumnos ayudantes en cada semestre. Además da la posibilidad de acceder a los planes de trabajo de los docentes en cualquier momento, siempre que hayan sido informatizados y gestiona la información referente a las visitas a clase que se realicen por parte de los departamentos a los profesores y alumnos ayudantes de los mismos (Verdecia Álvarez, y otros, 2007).

Características:

Utiliza PHP como lenguaje de programación Web Macromedia Dreamweaver, Smarty y NuSphere para hacer mucho más ágil, dinámico y con más calidad el desarrollo del sistema. Una vez concluido su desarrollo, no fue desplegado.

Después de todas las características antes expuestas sobre los sistemas nacionales, se puede concluir que ellos tampoco satisfacen las necesidades específicas del problema planteado. Además, la tecnología que se empleó para su desarrollo no se corresponde con las especificaciones tecnológicas para la Facultad 3 (SO3) que aparecen a continuación:

PLATAFORMA TECNOLÓGICA FACULTAD 3	
Para la inclusión de alguna aplicación:	<ul style="list-style-type: none"> • Servidor: Apache 2.6.x • Lenguaje de Programación: PHP 5.3.x • Bases de Datos: MySQL 5.1.x o PostgreSQL 8,4 o superior (Preferiblemente Postgres) • IDEs: Netbeans 6.9 o mayor, Eclipse, Aptana. • Frameworks: Symfony 1.4 o mayor, CodeIgniter 1.7.3 o mayor, Sauxe (preferiblemente no usar Sauxe, por la diferencia entre versiones de PHP que usan)
Metodologías de desarrollo:	<ul style="list-style-type: none"> • AUP (Agilup), XP, SCRUM, SXP
Para la inclusión de un sitio web:	<ul style="list-style-type: none"> • CMS Joomla (versiones 1.5 y 1.7, preferiblemente 1.7 para ganar en mayor seguridad, y soporte) • Para los blogs CMS Wordpress 3.x
Repositorios:	<ul style="list-style-type: none"> • Se decide montar el repo_f03 en el Repo Central UCI.
Generales:	<ul style="list-style-type: none"> • Frameworks de formularios: ExtJS v: 3,0 o mayor, JQuery v.1.8.x, Mootools v.1.3 • La comunicación entre todos estos sistemas que deben al final tributar al portal (Dragones), deben hacerse a través de Servicios Web a través del protocolo WSDL (Web Services Description Language). • Para el caso de los formularios electrónicos usar el Acrobat 8.0 PRO. • Para el firmado digital usar el Acrobat 8.0 PRO (y el de Linux)

Tabla 1 Plataforma tecnológica de la facultad 3. (Reyes, 2011)

Por lo tanto queda demostrada la necesidad de iniciar un nuevo proceso de desarrollo para darle solución al problema anteriormente planteado. Por consiguiente se hace necesario planificar lo necesario en pos de alcanzar el objetivo trazado, comenzando por las tecnologías, herramientas y metodologías.

1.4 Selección de las tecnologías, herramientas y metodologías

En la actualidad, se hace extensivo el uso de todo tipo de soluciones informáticas para facilitar el manejo y procesamiento de diferentes informaciones. El éxito en la construcción de dichas soluciones está relacionado con las metodologías, lenguajes y herramientas empleados para conducir eficaz y competitivamente su creación.

1.4.1 Metodologías de Desarrollo de Software

Durante el desarrollo de software, pueden aparecer diversos contratiempos o cambios en el negocio que impactan significativamente el proceso de ejecución de todas las actividades y la creación de los artefactos. Por lo que proveer al equipo de trabajo de un conjunto de procedimientos, técnicas, herramientas y soporte documental como medios para enfrentar dichas situaciones, puede facilitar la estructuración y dirección del desarrollo en función de la obtención de un resultado más acorde a las necesidades y expectativas de los clientes, así como alineado con las normas y prácticas internacionales para la producción de software. Para alcanzar tal fin, existen diversas metodologías clasificadas en dos categorías: *Ágiles* y *Tradicionales*, de las cuales a continuación se muestra una tabla comparativa:

Metodologías Ágiles	Metodologías Tradicionales
Basadas en Heurísticas provenientes de prácticas de producción de código.	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo.
Especialmente preparadas para cambios durante el proyecto.	Poseen cierta resistencia a los cambios.
Proceso menos controlado, con pocas normativas y restricciones.	Proceso mucho más controlado, con numerosas políticas/normas.
No existe contrato tradicional o al menos es bastante flexible.	Existe contrato prefijado.
El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Debido al tamaño del equipo de desarrollo, se necesitan pocos roles.	Debido al tamaño del equipo de desarrollo, existen numerosos roles especializados.
Pocos artefactos.	Muchos artefactos.
Menos énfasis en la arquitectura del software.	La arquitectura del software es esencial y se expresa mediante modelos.

Tabla 2 Diferencias entre las metodologías ágiles y las tradicionales. (TELLEZ, 2012)

Teniendo en cuenta que para la presente investigación, el equipo de desarrollo posee solo dos integrantes y que la planificación del cronograma de ejecución establece poco tiempo para darle

respuesta al problema en cuestión, la metodología de desarrollo a emplear será de tipo *Ágil*. Partiendo de un estudio por parte del proyecto y definición de la metodología de desarrollo, se determinó concebir la propuesta de solución de esta investigación con la metodología SXP, la cual será descrita a continuación.

1.4.1.3 SXP¹⁰

SXP es una metodología ágil y es un híbrido entre Scrum y XP, propuesta por desarrolladores cubanos de la Universidad de Ciencias Informáticas. Esta agrupa un conjunto de beneficios adquiridos de la unión de las mejores prácticas de las metodologías antes mencionadas y ofrece una estrategia tecnológica a partir de la introducción de procedimientos ágiles que permiten actualizar los procesos de software para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo y ayudando al líder del proyecto a tener un mejor control del mismo (Peñalver, y otros, 2010).

SXP consta de 4 fases principales:

- *Planificación-Definición*: Donde se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.
- *Desarrollo*: Es la fase donde se realiza la implementación del sistema hasta que esté listo para ser entregado.
- *Entrega*: Fase donde se pone en marcha el producto desarrollado y se hace la entrega al cliente.
- *Mantenimiento*: En esta se realiza el soporte para el cliente.

Está especialmente indicada para proyectos conformados por pequeños equipos y permite seguir de forma clara el avance de las tareas a realizar, facilitándoles a los líderes del desarrollo ver el progreso diario del trabajo. A continuación se muestran los artefactos que propone: Concepción del sistema, Modelo de Historias de Usuarios del Negocio, Lista de Reserva del Producto, Historias de Usuarios, Lista de Riesgos, Modelo de Diseño, Tarea de Ingeniera, Plan de Liberación, Estándar de Programación, Casos de Pruebas de Aceptación, Manual de Usuario, Manual de Identidad, Manual de Desarrollo, Gestión de Cambios.

¹⁰ Siglas adoptadas a partir de la fusión de las metodologías Scrum y XP: SXP.

Argumentación del uso de SXP:

Partiendo de la inminente renovación del staff del proyecto y de que ello implica una nueva capacitación sobre el negocio que está sujeto a cambios legislativos y de ejecución, se propone reforzar la documentación discreta que se obtendrá para garantizar que el conocimiento se comparta y se evite el estancamiento del desarrollo una vez los desarrolladores egresen de la Universidad (Peñalver, y otros, 2010). Por lo tanto, con basamento en el hecho de que la producción de software en la UCI ha sido certificada bajo normas y estándares internacionales, se pretende complementar el desarrollo de los entregables de la investigación siguiendo las pautas de SXP, con los siguientes artefactos validados y certificados en el expediente de proyecto propuesto para la producción¹¹,:

- 0128_Descripción de procesos de negocio.
- 0129_Descripción de requisitos.
- 0116_Reglas de Negocio.
- 0122_Diseño de Casos de Pruebas basado en Requisitos.

Facilitando entonces, el registro y conducción más organizados de toda la información del negocio y del desarrollo, donde se verán contenidos todos los artefactos propios de la metodología. Para hacer posible la modelación y representación de todos los elementos que propone esta metodología se hace indispensable el uso de herramientas que auxilien al equipo de desarrollo durante todas las etapas por las que transcurre.

1.4.2 Herramientas de Ingeniería de Software Asistida por Computadora¹² (Visual Paradigm 8)

Las herramientas CASE dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos de vida del desarrollo de un software. Por su gran relevancia, urge la necesidad de seleccionar una para dar asistencia en el progreso del sistema de gestión docente metodológico que se quiere llegar a efectuar.

Visual Paradigm es una herramienta con un enfoque Orientado a Objetos que soporta el ciclo de vida completo del proceso de desarrollo del software. Permite el aumento de la calidad a

¹¹ Ingeniería. Expediente de Proyecto del Proceso de Mejoras basado en CMMI2. Fuente: <http://calisoft.uci.cu>.

¹² Del inglés: Computer Assisted Software Engineering.

través de la mejora de la productividad en el desarrollo y mantenimiento del software, así como la reutilización del software, portabilidad y estandarización de la documentación. Soporta los principales lenguajes de modelado y normas industriales tales como Lenguaje de Modelado Unificado (UML) y BPMN¹³. Es multiplataforma y se integra con Eclipse / NetBeans / IntelliJ IDEA / Visual Studio. Brinda soporte a variedades de bases de datos como MySQL, MS SQL y Postgres (Visual-Paradigm, 1999-2013).

De acuerdo a las necesidades la herramienta CASE que más se ajusta es Visual Paradigm. Dicha solución va en correspondencia a que es multiplataforma, soporta la metodología seleccionada anteriormente y las aplicaciones Web. Es fácil de instalar y actualizar, además presenta varios idiomas. Tiene soporte de Mapeo de Objetos Relacionales (ORM)¹⁴, por lo cual se puede realizar transformaciones de diagramas de Entidad-Relación en tablas de base de datos. Durante el desarrollo de la investigación se hace indispensable visualizar cada requisito, para facilitar la comprensión del cliente. Por ello se hace necesario el uso de alguna herramienta que auxilie el desarrollo de los prototipos funcionales, en aras de una mejor interacción con el cliente, de ahí la necesidad del estudio de la herramienta Axure.

1.4.3 Herramienta para el desarrollo de prototipos de IU¹⁵ (Axure)

Cuando se realiza un software es difícil saber cómo los clientes piensan y se sienten con un diseño hasta que pueden visualizarlo. Muchas veces el equipo de desarrollo en la etapa final se siente frustrado y sin logros por las críticas negativas del cliente. Por ello surge la necesidad de crear prototipos de interfaz de usuario que sean mostrados antes de la liberación del producto, brindándole al cliente una idea de lo que será el resultado e inspirando mayor confianza al equipo de desarrollo al recibir buenas críticas.

Es una herramienta para crear prototipos y especificaciones muy precisas para páginas web. Permite construir una página web visualmente añadiendo, quitando y modificando los elementos con facilidad. Presenta una librería estándar que tiene varios elementos gráficos para el diseño, accediendo a gestionarlos, editarlos e incorporar nuevos componentes gráficos.

¹³ Del inglés: Business Process Modeling Notation.

¹⁴ Del inglés: Object Relational Mapping.

¹⁵ Del inglés: User Interface.

Admite exportar sus resultados de prototipado a Html, Word, Cvs¹⁶, algunos formatos de imagen (Bitmap, PNG, JPG o GIF) y RP que es el formato propio de Axure (además de la importación directa de este último) (Folch, 2012).

Sin embargo antes de realizar el prototipado se hace indispensable el modelado de los procesos para un mejor entendimiento de todas las actividades, datos y documentos que intervienen.

1.4.4 Notación de Modelado de Procesos de Negocio (BPMN)¹⁷

BPMN es una guía para representar cómo una empresa realiza sus objetivos centrales. Provee a las empresas entender sus procesos de negocio internos con el uso de gráficos, manteniendo así un estándar para comunicar estos procedimientos (Visual Paradigm, 1999-2013).

Esta notación de modelado de procesos de negocio proporciona una fácil comprensión por todos los usuarios del negocio. Crea un puente estándar entre el diseño y la implementación, llegando a estandarizar el modelo de procesos de negocios y notación proveyendo un medio sencillo de comunicar la información del proceso. Está diseñado para cubrir muchos tipos de modelos y permite la creación de procesos de negocio de extremo a extremo. Es por ello que para la notación de modelado del negocio BPMN es factible.

Sin embargo, se hace imprescindible el uso de un lenguaje para modelar el negocio. Tal es el caso de UML que presenta algunas características que lo hacen necesario, como es el caso de convertir los diagramas de procesos desarrollados con BPMN a diferentes tipos de diagramas que permiten un mayor entendimiento por los diferentes miembros del equipo de desarrollo, por ejemplo los diagramas de clases del diseño, entre otros.

1.4.5 Lenguaje Unificado de Construcción de Modelos (UML)¹⁸

En UML se representa un sistema o entidad del mundo real, donde simplifica los detalles relevantes del funcionamiento para facilitar su interpretación y sin ostentar ambigüedades ni

¹⁶ Del inglés: Comma Separated Values.

¹⁷ Del inglés: Business Process Modeling Notation.

¹⁸ Del inglés: Unified Modeling Language.

confusiones (López, y otros). Por la necesidad de representar el negocio para el entendimiento de los involucrados, urge seleccionar una herramienta de modelado para el desarrollo del sistema.

Permite captar la idea convencional del negocio, a especificar, visualizar y documentar los modelos de sistemas de software, incluyendo su estructura y diseño siendo fácil de comprender para comunicaras a otras personas (UML, 2003-2012). Es flexible, y se puede adaptar a entornos Orientado a Objetos (OO) o no, obtener uno dependiente de la plataforma o no como Ada o C (Atego, 2010-2013). Puede ser generador de código, e independiente de la metodología seleccionada (Object Management Group, Inc., 2012). Por presentar todas las particularidades ya mencionadas hacen de UML el lenguaje de modelado requerido para el desarrollo de la investigación. Dentro de sus aspectos se puede denotar que en gran medida es documentado, es un estándar abierto y puede estar orientado a objetos (Object Management Group, Inc., 2012). A pesar de que UML sirve también para modelar el negocio, se va a emplear únicamente para hacer los diagramas de clases, porque BPMN tiene un conjunto de características que favorecen en mayor medida la modelación del negocio. UML toma un enfoque orientado a objetos para el modelado, sin embargo BPMN tiene un enfoque orientado al proceso de modelado de sistemas. Mientras que UML se centra en el diseño de software, BPMN se enfoca en los procesos de negocio (Object Management Group, Inc., 2011). Además de que BPMN permite una representación gráfica con mayor número de especificaciones y diferenciaciones simbólicas. También la interpretación de sus modelos es más fácil de comprender por los desarrolladores y el cliente.

Una vez terminado el modelado y el diseño, es necesario iniciar la implementación de la propuesta. Con este objetivo se empleará como lenguaje de programación algunos de los que se integren con la versión 2 de SO3, además de las restantes herramientas, técnicas y métodos que serán usados. Tal es el caso de PHP.

1.4.6 Lenguajes de programación (PHP 5.3)

Un lenguaje de programación es aquella estructura que, con una cierta base sintáctica y semántica, imparte distintas instrucciones a un programa de computadora (MASTERMAGAZINE, 2007-2012). Igualmente pueden ser empleados para escribir páginas dinámicas y generarlas con rapidez. Tal es el caso del lenguaje interpretado PHP que es el

acrónimo de "**PHP: Hypertext Preprocessor**". Gran parte de su sintaxis fue tomada de C, Java y Perl con varias características propias de PHP (Mehdi Achour, 1997-2013).

PHP es un lenguaje de código abierto, fácil de aprender y permite crear páginas web dinámicas de manera rápida (The PHP Group, 2001-2013). Es multiplataforma y soporta la mayoría de servidores web y bases de datos. Brinda la posibilidad de utilizar programación por procedimientos, orientada a objetos, o una mezcla de ambas. Permite comunicarse con otros servicios usando protocolos como LDAP¹⁹, IMAP²⁰, SNMP²¹, NNTP²², POP3²³, HTTP²⁴, COM²⁵. Dicho lenguaje se integra con la herramienta CASE seleccionada (Visual Paradigm), es un software libre por lo cual facilita la obtención de actualizaciones y además soporta una amplia gama de gestores de bases de datos (Sklar, y otros, 2003). Debido a todos estos beneficios, será el lenguaje empleado para la construcción de la solución. Para proceder con tal fin, es necesario disponer de un marco de trabajo, por lo que en correspondencia con la tecnología definida por el proyecto, se describe Symfony a continuación en su versión 2.0.

1.4.7 Twig

Twig es un motor de plantillas para el lenguaje de programación PHP. El producto resultante es de código abierto se distribuye bajo licencia BSD. El marco de trabajo Symfony2 viene con un soporte incluido para Twig como su motor de plantillas por defecto. Además este compila las plantillas a código PHP sencillo y optimizado. También es flexible, ya que es impulsado por un léxico flexible y analizador. Esto permite al desarrollador definir sus propias etiquetas y filtros

¹⁹ Del inglés: Lightweight Directory Access Protocol (Protocolo Ligero de Acceso a Directorios).

²⁰ Del inglés: Internet Message Access Protocol es un protocolo de aplicación de acceso a mensajes electrónicos.

²¹ Del inglés: Simple Network Management Protocol (Protocolo Simple de Administración de Red).

²² Del inglés: Network News Transport Protocol (Protocolo para la Transferencia de Noticias en Red).

²³ Del inglés: Post Office Protocol (Protocolo de Oficina de Correo o Protocolo de Oficina Postal).

²⁴ Del inglés: Hypertext Transfer Protocol (Protocolo de Transferencia de Hipertexto).

²⁵ Es un dominio de Internet genérico alegórico a comercial, comerciante, comercio.

personalizados. En adición se puede usar independiente de Symfony2 ya que puede ser usado en cualquier proyecto PHP (Ecured, 2013).

1.4.8 Marco de trabajo

1.4.8.1 (Symfony 2.0)

Un marco de trabajo²⁶ ayuda a simplificar el desarrollo de un sistema mediante la utilización de distintos tipos de lenguajes, el uso de componentes, librerías y estructuras de acuerdo a los diferentes patrones existentes. También aporta agilidad en cuanto a la generación de código en algunos casos y la encapsulación de operaciones complejas en instrucciones sencillas. Todo en pos de que el desarrollador haga uso de las funcionalidades que el marco provea, como es el caso de Symfony.

Symfony²⁷ es un marco estable para desarrollar aplicaciones, está publicado bajo una licencia de software libre y es fácil de instalar y configurar en la mayoría de las plataformas. Es sencillo de usar en la mayoría de los casos y al mismo tiempo es flexible como para adaptarse a los casos más complejos (Potencier, y otros, 2008). De acuerdo a las necesidades del sistema y el lenguaje seleccionado, este marco es el ideal para el desarrollo. Fundamentado en que trabaja con PHP, es multiplataforma, se integra con proyectos de código abierto (Monolog, Doctrine, Assetic, Propel) y es compatible con la mayoría de gestores de bases de datos como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft (Potencier, 2007-2012). Hace uso del MVC (Modelo, Vista, Controlador) y sigue la mayoría de las mejores prácticas y patrones de diseño para la web (Potencier, y otros, 2008). Además se integra con la versión 2 en la cual está desarrollado el sistema SO3, por lo que con el uso de esta versión se puede llegar a mejorar el empleo y explotación de todas las nuevas bondades que tiene, como son: menos utilización de código, uso de las inyecciones de dependencias, aporte de rapidez y limpieza.

1.4.8.2 Bootstrap

Bootstrap es un marco de trabajo que permite crear interfaces web con CSS y Javascript que adaptan la interfaz dependiendo del tamaño del dispositivo en el que se visualice de forma

²⁶ Del inglés: Framework.

²⁷ Creado en 2005 por Fabien Potencier (presidente de Sensio (<http://www.sensio.com/>) empresa francesa de desarrollo de aplicaciones web).

nativa, es decir, automáticamente se adapta al tamaño de un ordenador o de una Tableta sin que el usuario tenga que hacer nada, esto se denomina diseño adaptativo o Responsive Design. El marco de trabajo trae varios elementos con estilos predefinidos fáciles de configurar: botones, menús desplegables y formularios, incluyendo todos sus elementos e integración con jQuery para ofrecer ventanas y tooltips dinámicos (OpenWebCMS, 2013). Por si solo un marco de trabajo no aporta todas las soluciones a las necesidades de un sistema. Dentro de las mismas se pueden evidenciar la compatibilidad de navegadores, aumentar las facilidades del trabajo con interfaces, entre otras especificidades.

1.4.9 Bibliotecas (Jquery 1.8)

En Internet se encontrarán muchas aplicaciones complejas por la incorporación de efectos visuales, arrastrar y soltar, auto-completar, animaciones, entre otras. Todas estas características en un sistema hacen aumentar la demora de la solución final por la necesidad de implementarlas desde cero. Las bibliotecas facilitan el trabajo con las interfaces y contribuyen en el desarrollo de la aplicación, además de auxiliar con la compatibilidad de navegadores, es por ello que se hizo necesario el estudio de una de ellas. Jquery²⁸, la cual es una de las bibliotecas de JavaScript para el desarrollo de aplicaciones. La misma mantiene compatibilidad con navegadores como Firefox, Internet Explorer, Safari, Ópera, Chrome y Konqueror, ya que unifica estándares CSS en diferentes navegadores (The Jquery Proyecto, 2010). Accede a los objetos del DOM²⁹ de un modo simplificado, ya que este provee una serie de métodos transversales de acceso para facilitar la selección de los objetos tanto de forma aleatoria como secuencial. (Castledine, y otros, 2010). Esta biblioteca ahorra muchas líneas de código y provee un conjunto de funciones para animar el contenido de la página de forma sencilla. Es rápida y concisa facilitando el uso del documento HTML, el manejo de eventos y la animación. Provee un mecanismo para la captura de eventos y maneja una gran cantidad de estándares para varios navegadores web.

Una vez descritos el lenguaje de programación, el marco de trabajo y la biblioteca a usar, es lógico pensar en cómo se gestionarán todos los datos con los cuales se trabaja en la realización de un sistema. El papel protagónico lo desempeñan los sistemas gestores de bases de datos.

²⁸ El autor de esta biblioteca es John Resig, actualmente trabaja para Mozilla Corporation.

²⁹ Del inglés: Document Object Model.

1.4.10 Sistema Gestor de Bases de Datos (PostgreSQL 9.1)

Un sistema gestor de bases de datos (DBMS³⁰) es un programa que permite definir una base de datos, elegir estructuras de información para el almacenamiento y la búsqueda de la misma. Es un modelo que facilita el acceso y describe los datos que serán almacenados permitiendo escribir algunas funcionalidades para manejarlos. Garantizan la confidencialidad, calidad, seguridad e integridad de los datos que contienen (SearchSQLServer.com, 2005-2013).

PostgreSQL es un sistema de base de datos de propósito general de gestión, en sus inicios fue llamado Postgres para finalmente adoptar el de PostgreSQL. Es libre para usar, modificar y redistribuir, es 100% software de código abierto y posee una licencia GPL³¹. Es una base de datos que se mantiene por mucho tiempo, y requiere poco o ningún mantenimiento en muchos casos. Presenta una arquitectura cliente-servidor, es altamente configurable y extensible (puede agregar sus propios tipos de datos, operadores, tipos de índices, y los lenguajes funcionales) para muchos tipos de aplicaciones y tiene una excelente escalabilidad y rendimiento.

El Sistema Gestor de Bases de Datos PostgreSQL tiene una documentación muy bien organizada, pública y libre. Es robusto, de alta calidad, fácil de mantener y con un código bien comentado. Al tener la mayoría de las herramientas para generar toda la información necesaria solo resta seleccionar una para comenzar la implementación.

1.4.11 Entorno de Desarrollo Integrado³² (Netbeans 7.2)

La interacción hombre-máquina para el desarrollo de aplicaciones es facilitada por los entornos de desarrollo integrados en aras de lograr un máximo beneficio. Permiten compilar el código generado por el programador, agregar elementos (clases, archivos, componentes), ayuda, cargar, salvar archivos y aporta opciones de compilación línea a línea. Además brinda la facilidad de tener una interfaz visual mediante los formularios del código implementado y facilita configurar las características en todo momento del objeto seleccionado. Debido a su gran importancia en el desarrollo de esta investigación se hace indispensable la selección y uso de una herramienta de este tipo.

³⁰ Del inglés: Data Base Management System.

³¹ Del inglés: PostgreSQL License.

³² Del inglés: Integrated Development Environment (IDE).

NetBeans es un IDE para que los programadores puedan escribir, compilar, depurar y ejecutar programas, es una herramienta de código abierto escrito en Java y fundado por Sun Microsystems en junio del 2000. Su lenguaje de nacimiento no es un impedimento para que pueda servir con cualquier otro lenguaje de programación. Existen dos variantes de código abierto y gratuitos: el NetBeans IDE (presenta un número importante de módulos para extender) y el NetBeans Platform (base modular y extensible usada como estructura de integración para crear grandes aplicaciones de escritorio). Es libre y gratuito, por lo cual se denota el incremento de extensiones adicionales por parte de empresas independientes asociadas, especializadas en desarrollo de software. En cuanto a la creación de aplicaciones profesionales de escritorio, empresariales, web y aplicaciones móviles, proporciona las herramientas necesarias para trabajar con el lenguaje Java, así como PHP, JavaScript, Groovy y Grails y C / C + +. Su rendimiento es mejorado (Oracle Corporation, 2012).

Al ser Netbeans un entorno de desarrollo que presenta suficiente documentación, herramientas de apoyo y un rendimiento estable, es el IDE ideal para el desarrollo de la investigación. Además, es multiplataforma, compatible con el lenguaje seleccionado en la investigación de código abierto y gratuito. Da soporte para múltiples marcos de trabajo PHP.

CONCLUSIONES

A partir del estudio de las soluciones internacionales y nacionales se evidenció que ninguna ofrece una solución al problema planteado por la investigación. Las diferentes herramientas con las que fueron desarrolladas no se integran completamente con las especificidades para el desarrollo de software del proyecto y algunas son privativas. Por lo tanto, considerando lo establecido por el proyecto y la investigación realizada por los autores se seleccionó como metodología de desarrollo a SXP y se propone complementar el registro documental con artefactos certificados de acuerdo al nivel 2 de CMMI. La herramienta CASE será el Visual Paradigm y como notación y lenguaje de modelado, BPMN y UML respectivamente. Asimismo se propone la construcción de los prototipos de interfaz de usuario usando el Axure. El desarrollo de la solución contará con el lenguaje de programación PHP, el marco de trabajo será Symfony y el gestor de base de datos PostgreSQL. Además se empleará la librería JQuery por todos los beneficios mencionados anteriormente.

CAPÍTULO 2: CONSTRUCCIÓN DE LA PROPUESTA

2.1 Introducción

En la construcción de la propuesta, se modelarán y describirán los procesos de manera que el negocio sea comprendido y representado cabalmente por los autores de la investigación. Además se mostrará la lista de requisitos funcionales, los conceptos y principales entidades identificadas, así como una propuesta de representación visual (prototipo) de las funcionalidades a desarrollar. Una de las mejores vías para comprender la evolución desde la investigación hasta la implementación, es la explicación de un caso de estudio.

2.2 Descripción del flujo actual de los procesos

Es preciso destacar que de todos los procesos docentes metodológicos, esta investigación solo se centrará, como se expresara anteriormente, en los procesos Guardia docente, Control a clases y Plan de trabajo metodológico. Además se tienen los subprocesos:

2.2.1 Proceso de gestión Guardia Docente

Breve descripción	El proceso inicia al realizarse la distribución de profesores para la guardia docente. A partir de lo cual, cada profesor asignado debe en el horario que le corresponde, controlar el cumplimiento de los reglamentos asociados al proceso docente-educativo y velar por la disciplina de los docentes y discentes durante el período de clases.
Involucrados	Vicedecana de formación, profesores designados.
Entradas/salidas	Distribución de la guardia docente.pdf Descripción de la guardia docente.doc

2.2.2 Proceso de gestión de los Controles a clases

Breve descripción	El proceso inicia al designar la realización de un control a clases de un profesor. Para el mismo los controladores deben tener presente un conjunto de indicadores que, finalmente, tributarán al registro de una nota cualitativa. Uno de los elementos más importantes de este proceso es el registro de las deficiencias y principales logros del docente durante la impartición de una clase.
Involucrados	Decano, vicedecanos, jefes de departamento y de asignatura.

Entradas/salidas	Plan de Controles a clases.xls Planilla de Control a Clases.doc
------------------	--

2.2.3 Proceso de gestión Plan de trabajo metodológico

Breve descripción	El proceso inicia cuando diversas orientaciones desde el nivel central llegan al vicedecanato de formación, donde se elabora el Plan de trabajo metodológico de la facultad y luego se distribuyen dichas orientaciones a cada uno de los Departamentos docentes, de manera que el trabajo metodológico de todos los profesores esté alineado con los objetivos y líneas trazadas por la Universidad y la Facultad.
Involucrados	Vicedecana de formación, jefes de departamento, asesores de año y jefes de asignatura.
Entradas/salidas	Plan de trabajo metodológico UC1.doc Plan de trabajo metodológico F3.doc Plan de trabajo metodológico DD.doc Plan de trabajo metodológico A.doc

Al haber documentado todos los procesos, se tiene mayor manejo de todos los conceptos fundamentales y de cómo se establece su interacción. Para que el conocimiento de los mismos sea compartido con todo el personal involucrado y permita al equipo de proyecto mayor facilidad en el manejo de las siguientes tareas, es preciso realizar un mapa conceptual.

qué se genera y qué son. Este mapa conceptual brinda información básica e inicia algunas ideas a tener en cuenta en el Modelo de Procesos donde se modelará el funcionamiento, los involucrados y todos los documentos que se crean, modifican y envían.

2.4 Modelo de proceso del negocio Plan de trabajo metodológico

En un desglose del modelado, para lograr comprender a cabalidad el negocio, se toma como caso de estudio al proceso Plan de trabajo metodológico, del cual se expondrá a continuación el modelado de la secuencia de actividades del mismo. El resto de los artefactos podrán ser consultados en el expediente de proyecto de SO3.

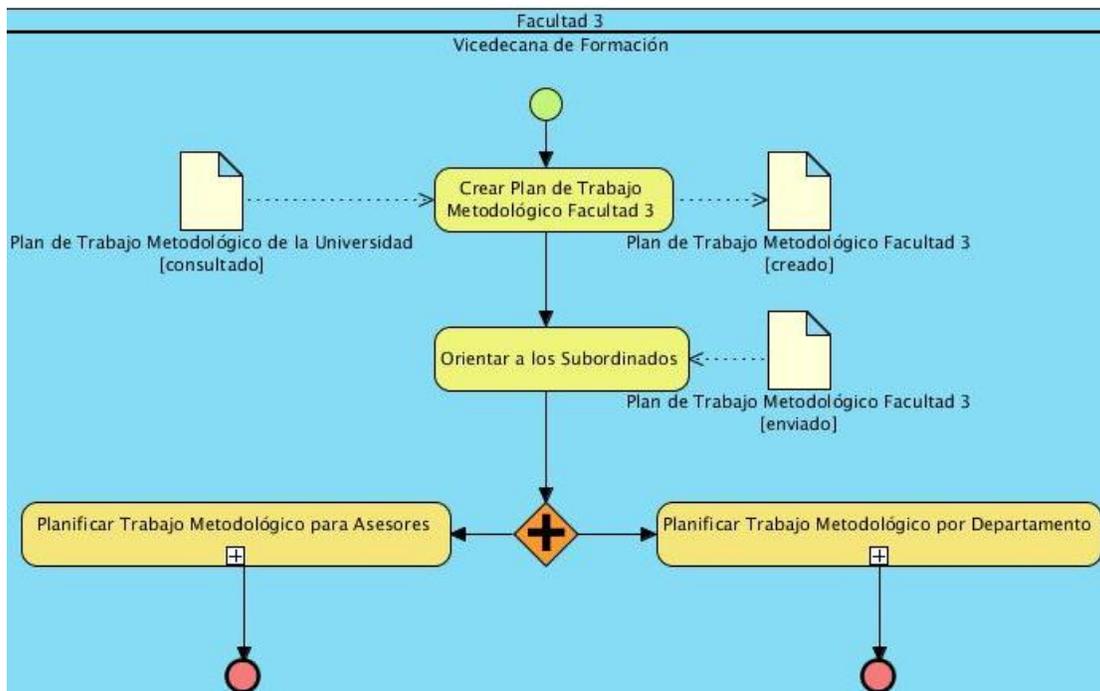


Figura 2 Diagrama de Procesos de Negocio: Plan de trabajo metodológico.

Para una mejor comprensión del funcionamiento del proceso y la realización del diagrama de Plan de trabajo metodológico se explicará a continuación el subproceso Planificar Trabajo Metodológico por Departamento.

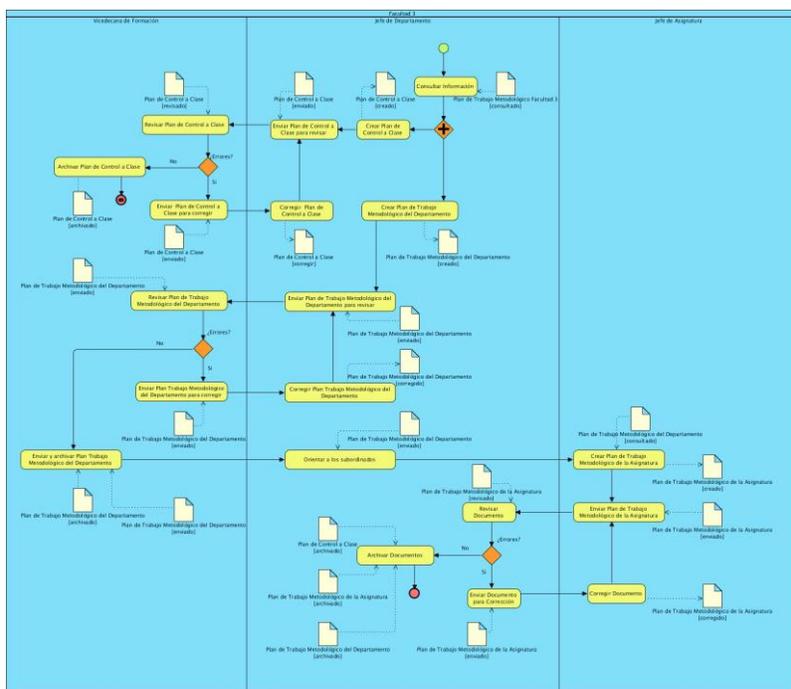


Figura 2.1 Plan de trabajo metodológico: Planificar Trabajo Metodológico por Departamento.

Para un mejor entendimiento del diagrama se desglosa la imagen de la siguiente manera:

En la *Figura 2.1.a* se puede apreciar el inicio del flujo del proceso de negocio Plan de trabajo metodológico. Donde la primera actividad es consultar la información del Plan de trabajo metodológico de la Facultad 3 para conocer las orientaciones. Las dos siguientes actividades, Crear Plan de control a clases y Crear Plan de trabajo metodológico del Departamento se ejecutan concurrentemente, permitiendo la creación de los documentos Plan de control a clases y Plan de trabajo metodológico del Departamento. Ambos documentos serán enviados a la vicedecana de formación para revisarlos donde se ejecutan las actividades diagramadas en la *Figura 2.1.b*.

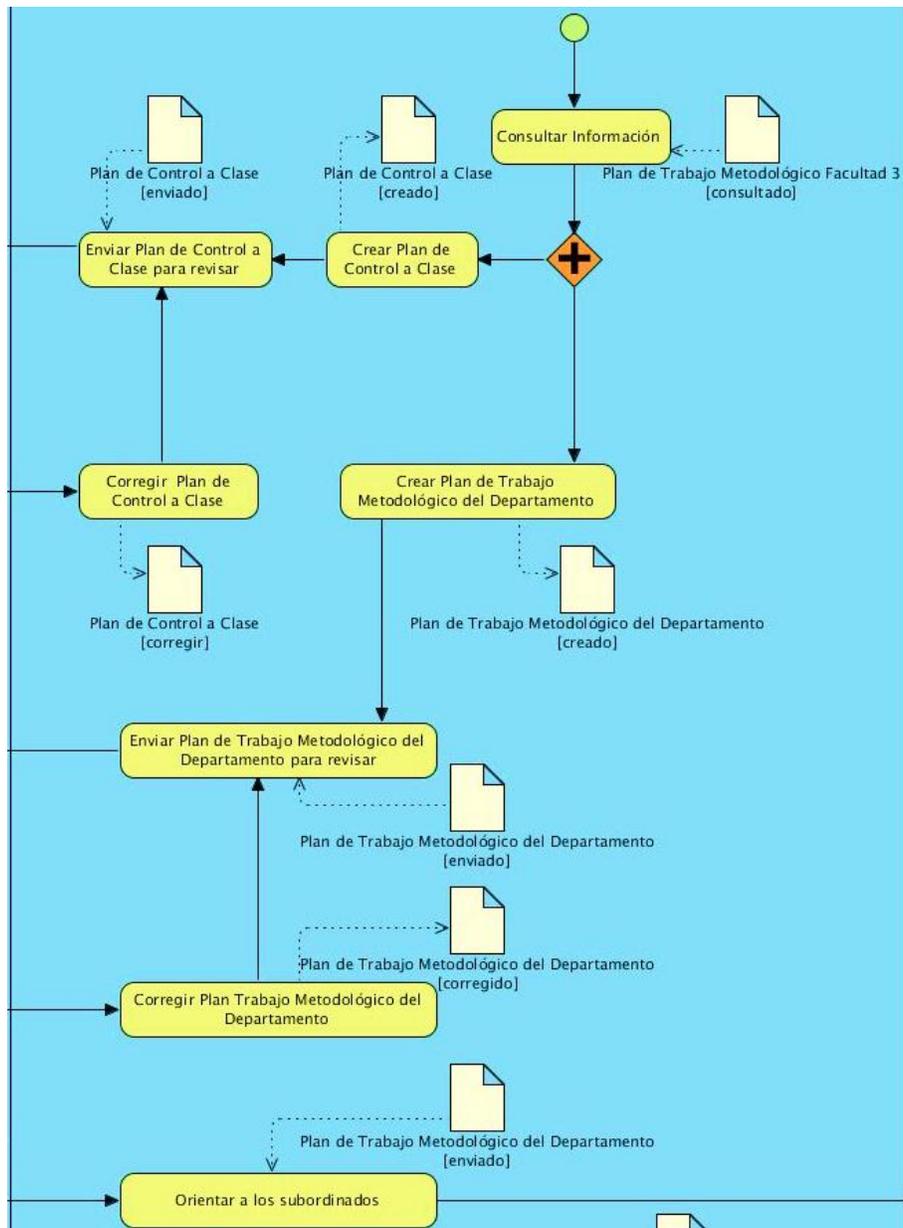


Figura 2.1.a Pool: Jefe de Departamento.

La vicedecana al recibir el Plan de control a clases y el Plan de trabajo metodológico del Departamento los revisa, sino no contienen errores, archiva ambos y envía al jefe del departamento el Plan de trabajo metodológico del Departamento para que éste lo archive. En caso de estos presentar errores, envía ambos documentos al jefe del departamento para que los corrija, una vez corregidos, se envían nuevamente para su revisión. Estas actividades se

vuelven a realizar tantas veces como existan errores. Luego se ejecutan las actividades diagramadas en la *Figura 2.1.c*.

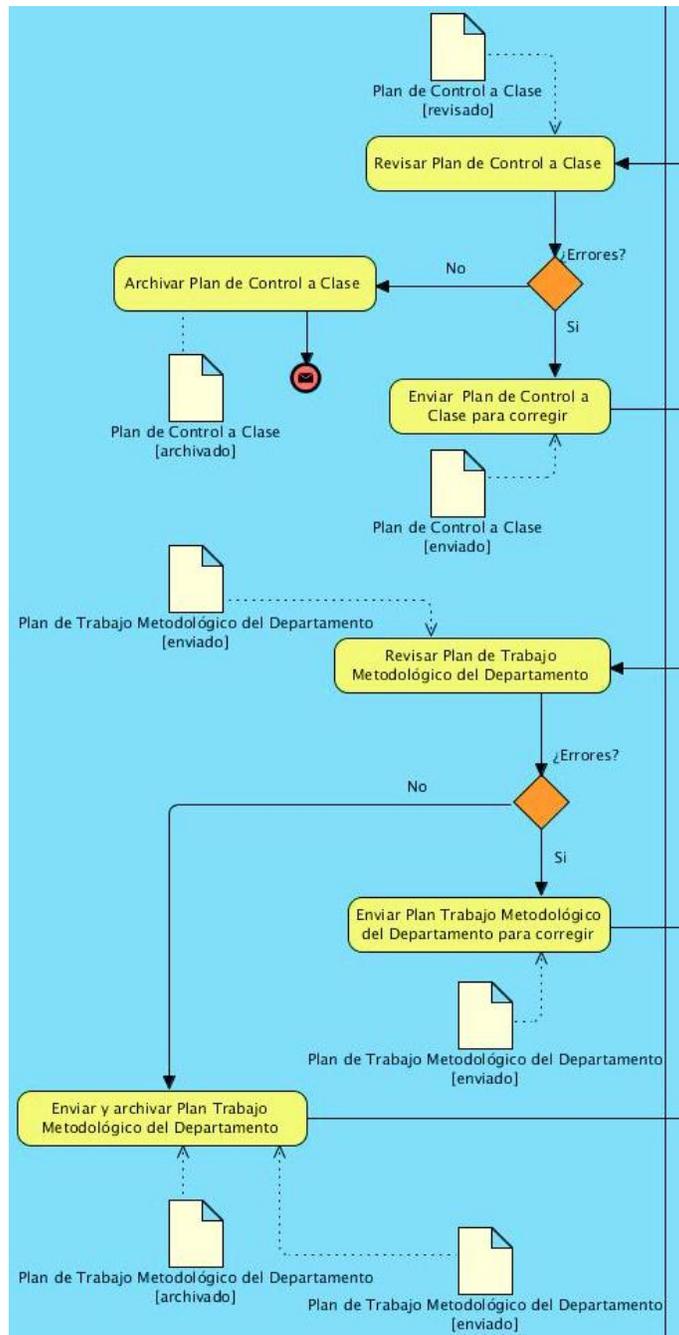


Figura 2.1.b Pool: Vicedecana de Formación.

Después de estar correcto el Plan de trabajo metodológico del Departamento, se le envía a sus subordinados, los jefes de asignatura, lo consultan y crean el documento Plan de trabajo

metodológico de la Asignatura. Luego lo envían el jefe del departamento para su revisión, si presenta errores le envía el documento Plan de trabajo metodológico de la Asignatura al jefe de asignatura para su corrección. Después de corregir el documento, se envía nuevamente el Plan de trabajo metodológico de la Asignatura al jefe de departamento, estas actividades se vuelven a realizar tantas veces como existan errores. Finalmente todas las actividades confluyen hacia la actividad Archivar Documentos.

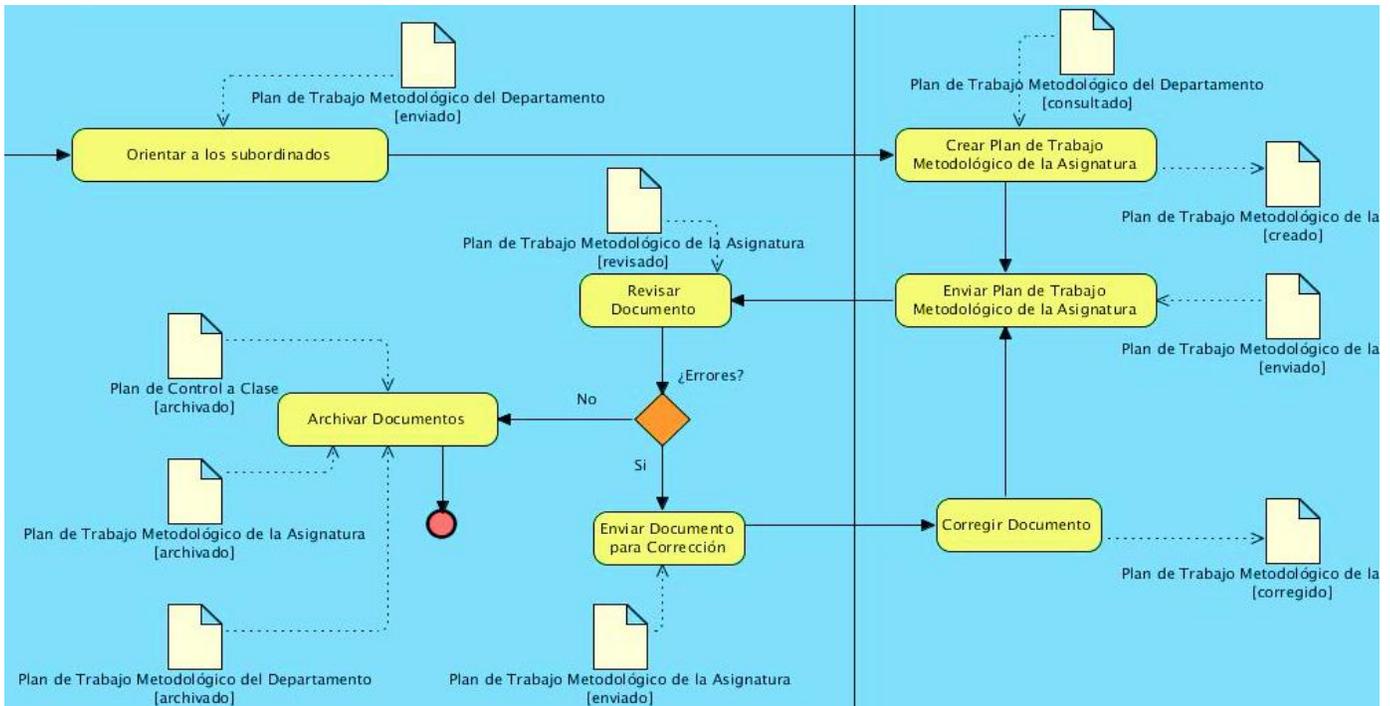


Figura 2.1.c Pool: Jefe de Asignatura.

Todas estas actividades y documentos se integran para permitir el funcionamiento general del negocio, pero para poder tener la base para su posible informatización, es necesario realizar la descripción de los mismos para documentarlos y llegar a entender su correcta ejecución.

2.5 Descripción del proceso del negocio Administrar Plan de trabajo metodológico

Objetivo	<ol style="list-style-type: none"> 1. Planificar un conjunto de actividades relacionadas con las líneas de trabajo metodológico, de manera que se les dé cumplimiento a los objetivos trazados y solución a los problemas identificados. 2. Recopilar toda la información referente al trabajo metodológico en todos los niveles.
-----------------	---

	<ol style="list-style-type: none"> 3. Regir e integrar de manera sistemática las actividades que se realizan en todos los niveles y estructuras relacionadas con el proceso docente educativo. 4. Proveer a los docentes de un espacio metodológico para su preparación y superación en diferentes temas relacionados con el proceso de formación. (1-2) 5. Enfocar la preparación de los docentes en el perfeccionamiento del Modelo de Formación centrado en el Aprendizaje.
Evento(s) que lo genera(n)	<ol style="list-style-type: none"> 1. Necesidad de organización, superación y planificación de las estructuras involucradas en el proceso de formación docente. 2. Existencia de un banco de problemas metodológicos enfocados a la docencia. 3. Plan de acciones para la preparación de los docentes con menor experiencia de trabajo y mayores problemas metodológicos. 4. Identificación de las líneas de trabajo y los objetivos generales y específicos.
Pre condiciones	<ol style="list-style-type: none"> 1. Deben estar definidos los objetivos generales y específicos. 2. Debe haberse identificado un banco de problemas metodológicos para darle solución. 3. Deben haberse definido las líneas de trabajo metodológico.
Marco legal	<ol style="list-style-type: none"> 1. Reglamento Docente y Metodológico. Resolución No. 210/07, Capítulo II. Trabajo Metodológico Artículo 26. 2. Reglamento Docente y Metodológico. Resolución No. 210/07, Capítulo II. Trabajo Metodológico Artículo 42. 3. Reglamento Docente y Metodológico. Resolución No. 210/07, Capítulo II. De las formas y tipos del trabajo metodológico: Artículo 44.
Clientes internos	<ol style="list-style-type: none"> 1. Departamentos docentes 2. Colectivos de asignatura 3. Vicedecanato 4. Decanato
Clientes externos	N/A
Entradas	Plan de trabajo metodológico (Universidad)

Flujo básico Administrar Plan de trabajo metodológico		
1.	Crear Plan de trabajo metodológico Facultad 3: se crea el documento a partir de consultar el Plan de trabajo metodológico de la Universidad, el cual será la referencia para crear el de la facultad.	
2.	Orientar a los Subordinados: luego de creado el Plan de trabajo metodológico de la Facultad 3 se le envía a los subordinados, para que confeccionen el propio.	
3.	Planificar Trabajo Metodológico por Departamento: en este subproceso se encuentran todas las actividades relacionadas con los departamentos y sus asignaturas que se realizan de manera diferente a los asesores de año.	
Pos-condiciones		
1.	N/A	
Salidas		
Plan de trabajo metodológico de la Facultad 3 (PDF).		
Flujos paralelos		
1.	3. a Planificar Trabajo Metodológico para Asesores.	Flujo de actividades que culminará con la creación del Plan de trabajo metodológico de Asesoría de Año.
Pos-condiciones		
N/A		
Salidas		
1. Plan de trabajo metodológico de Asesoría de Año (PDF).		
Asuntos pendientes		
N/A		

Después de realizadas las descripciones de los procesos, se debe obtener el listado de las funcionalidades para la construcción de la propuesta, para ello deben aplicarse diferentes formas de recolección de la información.

2.6 Técnicas de recopilación de requisitos

En el proceso de desarrollo del sistema de gestión docente metodológica el equipo de desarrollo se enfrenta a la identificación de requisitos, de manera que resulta importante entender lo que el cliente quiere, analizar sus necesidades, especificar una solución sin ambigüedades y administrar los requisitos conforme a su transformación en un sistema operacional (Pressman, 2010). Para ello existen diversas técnicas de captura de información que facilitan la comunicación y la extracción de los elementos más esenciales del negocio. Las técnicas seleccionadas para realizar la captura de requisitos fueron:

Entrevistas: desarrollada a través de encuentros con el cliente para poder establecer los objetivos del sistema, qué es lo que debe lograr y de qué forma satisfacer las necesidades del negocio.

Tormentas de ideas: realizada mediante encuentros del equipo de trabajo, de modo que tanto desarrolladores como clientes aportaron ideas enriqueciendo el conocimiento de los implicados, facilitando la obtención desde diferentes puntos de vista y razonamientos de la información inherente a la investigación.

Mapa conceptual: empleada para favorecer la representación visual de todos los conceptos que intervienen en el negocio, de manera que esta primera entrega contribuya con la creación de la base de datos.

Modelación: técnica empleada para lograr la representación a través de símbolos, del flujo de actividades de los procesos, permitiendo la identificación de todos los pasos lógicos y restricciones del negocio.

2.6.1 Requisitos funcionales

Después de realizada las técnicas de captura de requisitos se tienen las necesidades del negocio bien detalladas y descritas. Las funcionalidades que serán desarrolladas en la propuesta de solución, se listan conformando la siguiente enumeración de requisitos inherentes a la presente investigación:

R1 Administrar Plan de control a clases

R1.1 Crear Plan de control a clases

R1.2 Modificar Plan de control a clases

R1.3 Eliminar Plan de control a clases

R2 Guardar Plan de control a clases

R3 Administrar Planificación de Controles a clases

R3.1 Crear una Planificación de Control a Clases

R3.2 Modificar una Planificación de Control a Clases

R3.3 Eliminar una Planificación de Control a Clases

R4 Guardar una Planificación de Control a Clases

R5 Administrar Planilla del Control a Clase

R5.1 Crear Planilla del Control a Clase

R5.2 Modificar Planilla del Control a Clase

R5.3 Eliminar Planilla del Control a Clase

R6 Guardar Planilla del Control a Clase

R7 Generar reportes

- R7.1** Profesor evaluado
- R7.2** Controlador
- R7.3** Categoría docente
- R7.4** Asignatura
- R7.5** Tipo clase
- R7.6** Evaluación de los indicadores
- R7.7** Evaluación
- R7.8** Departamento
- R8** Administrar Plan de trabajo metodológico
 - R8.1** Crear Plan de trabajo metodológico
 - R8.2** Modificar Plan de trabajo metodológico
 - R8.3** Eliminar Plan de trabajo metodológico
- R9** Generar reportes
 - R9.1** Actividades de una línea de trabajo
 - R9.2** Responsable
 - R9.3** Participantes
 - R9.4** Estado de las actividades
 - R9.5** Reporte estadístico del Plan de trabajo metodológico (días 20)
- R10** Modificar estado de las actividades
- R11** Adicionar profesores al listado de la guardia docente
- R12** Eliminar profesores del listado de la guardia docente
- R13** Administrar Distribución de la Guardia Docente
 - R13.1** Crear Distribución de la Guardia Docente
 - R13.2** Eliminar Distribución de la Guardia Docente
- R14** Guardar Distribución de la Guardia Docente
- R15** Administrar Descripción de la Guardia Docente
 - R15.1** Crear Descripción de la Guardia Docente
 - R15.2** Modificar Descripción de la Guardia Docente
 - R15.3** Eliminar Descripción de la Guardia Docente
- R16** Guardar Descripción de Guardia Docente
- R17** Generar reportes
 - R17.1** Nombre del controlador
 - R17.2** Incidencias

- R17.2.1** Horario de clases
- R17.2.2** Nombre del Profesor
- R17.2.3** Grupo
- R17.2.4** Asignatura

R18 Administrar Notificaciones

- R18.1** Crear notificación
- R18.2** Modificar notificación
- R18.3** Eliminar notificación

R19 Administrar departamentos

- R19.1** Crear departamento
- R19.2** Modificar departamento
- R19.3** Eliminar departamento

R20 Administrar asignaturas

- R20.1** Crear asignatura
- R20.2** Modificar asignatura
- R20.3** Eliminar asignatura

R21 Determinar semanas

- R21.1** Crear semanas
- R21.2** Modificar semanas
- R21.3** Eliminar semanas

Para cada una de estas funcionalidades, se construyó un Prototipo de interfaz de usuario (PIU), con el objetivo de familiarizar al cliente con la propuesta del equipo de desarrollo. De esta comunicación hombre-aplicación es posible perfeccionar la concepción que se tiene del negocio durante la interacción del cliente con los PIU navegables diseñados en Axure.

2.6.2 Prototipo de Interfaz de Usuario

Los PIU permiten tener una noción de las funcionalidades a ser representadas gráficamente en las Interfaces de usuario. Su diseño pretende preparar al cliente para enfrentar el uso del sistema final y apreciar a partir de los primeros ensayos, las preferencias del cliente, en función de ofrecer un diseño que centre su valor en la satisfacción de las expectativas del mismo.

El prototipo siguiente representa la primera imagen a visualizar del crear Plan de trabajo metodológico (PTM). En esta se seleccionará el curso escolar en el que se encuentra y el nivel

al que pertenece dicho plan ya sea de la Facultad 3, de algún departamento, asignatura o asesores de año. Se muestra además la restante información que se observará en el prototipo de este requisito que se evidencia a continuación.

Crear plan de trabajo metodológico

Semestre: Área:

Fundamentación

Problemas metodológicos

Indicadores

Línea Metodológica	Nivel	Objetivos
<input type="text" value="Seleccione la línea para ver sus actividades"/> <input type="text" value="Seleccione"/>		

Otras actividades

Actividad	Objetivos
<input type="text"/>	<input type="text"/>

Figura 3 Prototipo: Crear Plan de trabajo metodológico.

Estos prototipos permiten tener en mente muchos de los detalles que deben tenerse en cuenta para la creación de un sistema. Sin embargo, esto no es suficiente para tener la base y poder implementar. Pues se hace necesario esbozar todas las clases con sus atributos, las cuales darán el punto de partida para conformar las tablas donde se contendrán todos los datos. La manera de representar esto, además de sus posibles relaciones es el Modelo de Entidad Relación.

2.7 Modelo Entidad Relación (MER)

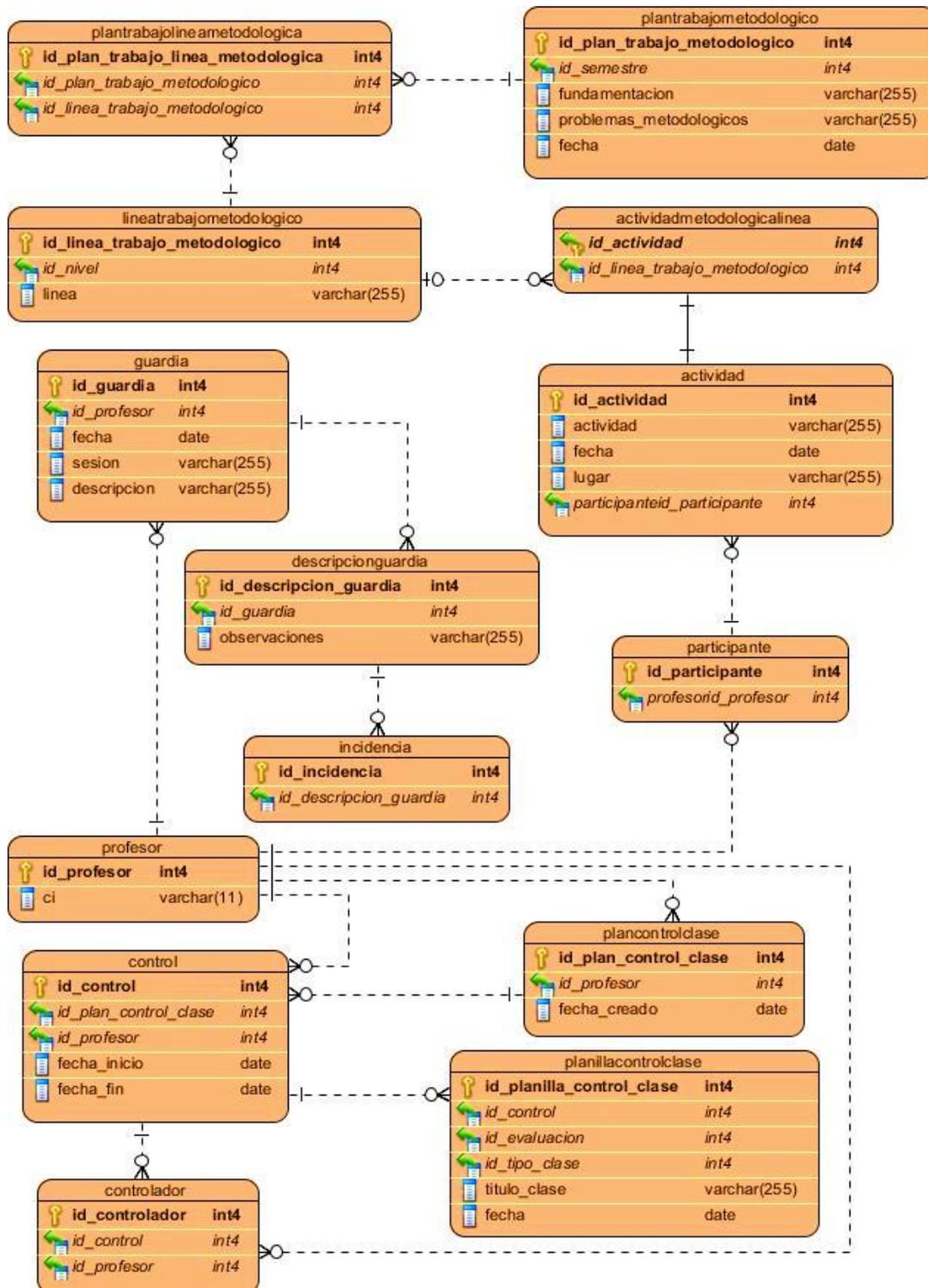


Figura 4 Modelo Entidad Relación de las clases significativas.

El MER de la presente investigación consta de treinta y cuatro tablas del negocio, de ellas catorce son entidades significativas. Dichas entidades significativas son las fundamentales para el sistema, las mismas son las bases para el manejo de los principales datos de cada proceso. Para su funcionamiento e integración con SO3, hace uso de seis tablas nomencladoras. La base de datos está en tercera forma normal para garantizar la integridad de los datos, un correcto diseño y una mayor consistencia de los datos.

Las bases de datos, por si solas no brindan una estructura potencialmente organizada y distribuida para la programación, es por ello que el estudio de los patrones a utilizar, resulta de vital importancia.

2.8 Patrones

En la construcción de software pueden surgir problemas específicos y que pueden ser recurrentes en el mismo escenario o en otros. Por lo tanto la solución a ellos debe ser efectiva y reusable, así como debe facilitarle al software ser más resistente al cambio, ayudar a especificar interfaces, reutilizar código y usar documentación estándar. Dicha solución es conocida como un patrón de software (Canal, 2005).

2.8.1 Patrón arquitectónico Modelo Vista Controlador (MVC)³³

El patrón MVC es el encargado de separar la lógica de negocio, de la interfaz (vista) del usuario, facilitando con ello una mejor funcionalidad, mantenibilidad y escalabilidad del sistema. Esta característica hace que los sistemas sean de forma simple, sencilla y que en un mismo código no estén mezclados varios lenguajes de programación (Bahit, 2009). MVC aumenta la reutilización y desacopla vistas y modelos mediante el establecimiento de un protocolo de suscribirse/notificar entre ellos. El empleo de este patrón arquitectónico, favorece el uso de otros patrones como Fachada, Controlador, Comando, entre otros.

Con la aplicación de este patrón, las vistas, el modelo y los controladores se pueden construir por separado y luego unirlos en tiempo de ejecución (porque existe una clara separación entre los componentes). Si un componente funciona mal, puede reemplazarse sin afectar a las demás. La relación entre el modelo y sus vistas es dinámica (Gamma , y otros, 1995). El marco de trabajo Symfony2 está basado en MVC. En general simplifica el proceso tomando lo mejor

³³ Del inglés: Model View Controller.

de la arquitectura MVC y la implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo. El patrón divide las aplicaciones en tres niveles de abstracción:

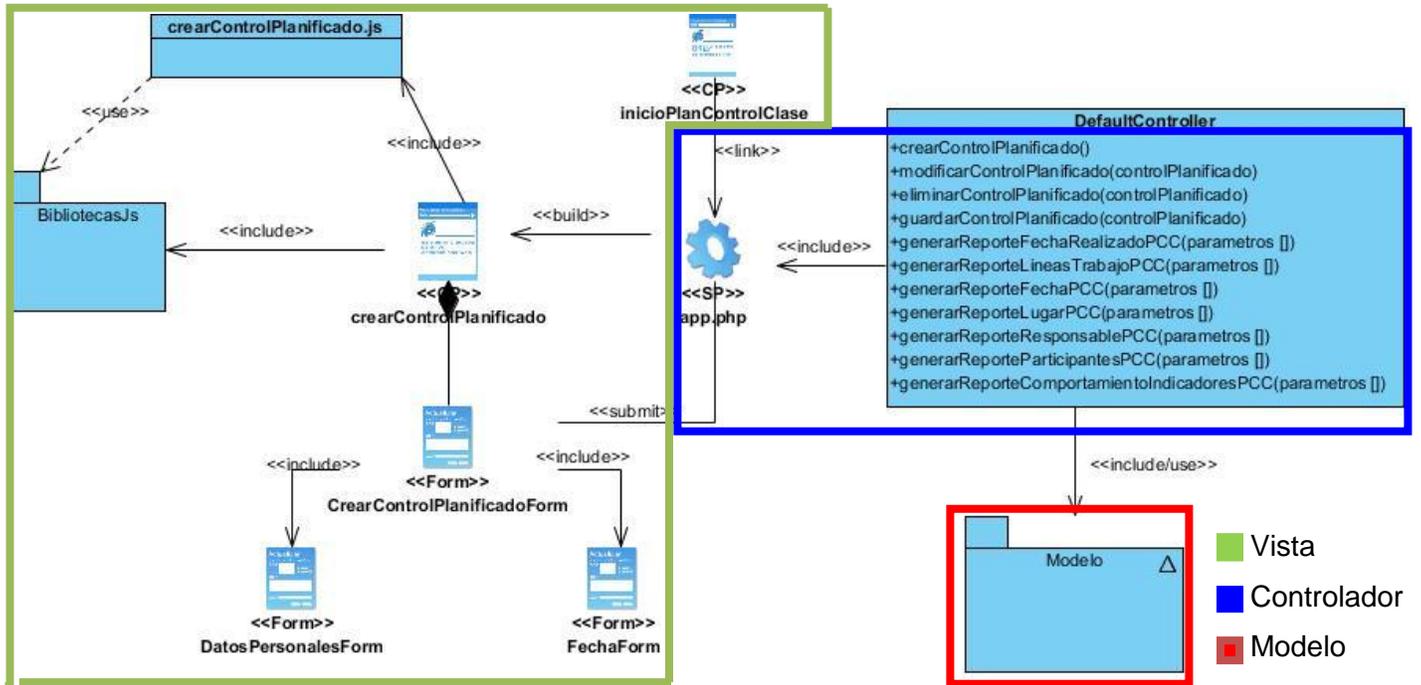


Figura 5 Patrón Modelo Vista Controlador en Diagrama de Clases del Diseño del requisito Crear Planificación de Control a Clase.

Luego de crear todos los paquetes necesarios para la implementación, Symfony2 crea una estructura de carpetas representando el MVC. A continuación se muestran algunas aplicaciones de este patrón (Potencier , y otros, 2008):

- **Controlador frontal:** El controlador frontal es generado por el marco de trabajo y trae consigo el código relativo al MVC. El mismo se encarga de decodificar la petición y transferirla a la acción correspondiente.

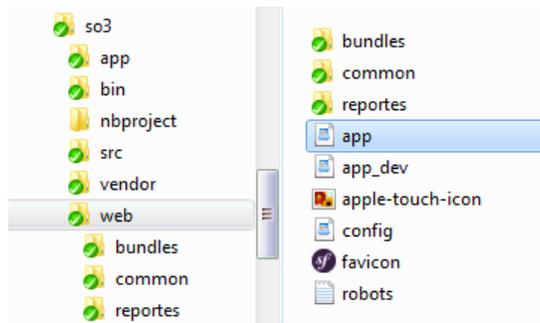


Figura 6 Uso del controlador frontal de la Plataforma Informativa SO3.

En este nivel se encuentran las clases controladoras, las cuales en Symfony2 se nombran con el sufijo Controller. Las clases controladoras incluyen el código específico del paquete al que pertenecen, verifican la integridad de las peticiones y preparan los datos requeridos por la vista (pueden hacer uso de otros componentes para obtener la información y generar las páginas).

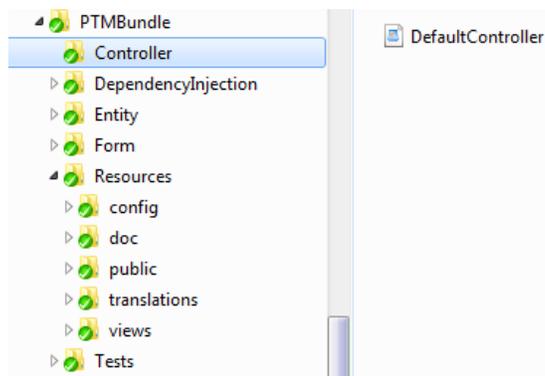


Figura 7 Paquete PTMBundle donde se evidencia el uso de los controladores.

Vista: Los usuarios utilizan para interactuar con la aplicación, las vistas. Éstas son páginas creadas mediante una plantilla (usando los gestores de plantillas) y que inserta los datos brindados por el Controlador después de gestionarlos desde el Modelo.

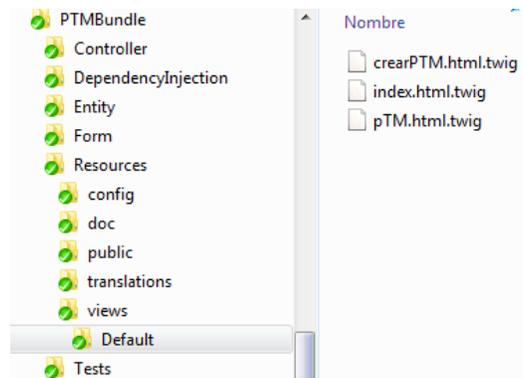


Figura 8 Paquete PTMBundle donde se evidencia el uso de las vistas.

En este nivel se encontrarán todas las páginas a visualizar por la persona que interactúe con el sistema. Las cuáles harán uso de la biblioteca JQuery ya mencionada en el capítulo 1, Bootstrap para una mejor visualización de cada elemento en las interfaces, además de JavaScript y CSS. Todas las vistas son desarrolladas con Twig como motor y lenguaje de plantillas. Ya que permite una sintaxis concisa y limpia, por lo que son fáciles de leer y de escribir (Sensio Labs, 2010-2012).

- **Clases de la capa del modelo:** Los ORM se encargan de generar las clases de la capa del modelo (en función de la estructura de datos de la aplicación se crea la estructura básica de las clases) y genera automáticamente el código necesario. Dichas clases son las tablas de la base de datos traducidas a clases PHP que Symfony2 las denomina entidades del sistema. Las mismas permiten interactuar con la Base de Datos en sustitución a las sentencias SQL mediante un objeto llamado *Entity Manager* (Erguiluz, 2012).

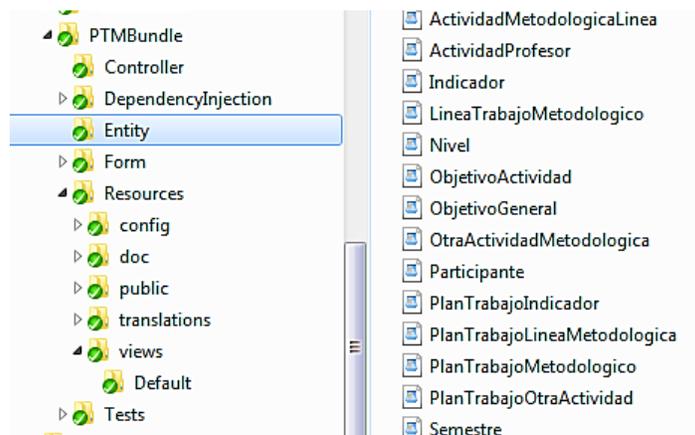


Figura 9 Paquete PTMBundle donde se evidencia el uso de las entidades.

PDO (*PHP Data Objects*) garantiza un estándar para la realización de las consultas a la base de datos sin importar el gestor que se utilice. Para la manipulación de la información en la Base de Datos se hace uso de Doctrine2 a través del *Entity Manager*.

2.8.2 Otros patrones de Software

A parte de los patrones mencionados previamente, se tienen los de asignación General de Responsabilidad (GRASP)³⁴ los cuales describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades y los GoF³⁵, para controlar la creación (preocupación con el proceso de creación de objetos y clases) de estructuras (composición de las clases y objetos) y sus comportamientos (caracterizan la interacción y la responsabilidad de los objetos y clases) en el software. Estos patrones son de gran importancia porque influyen en la solidez, la capacidad de mantenimiento, la reutilización de código y la obtención de los objetos o las abstracciones adecuadas (Larman, 1999).

El marco de trabajo por si solo utiliza algunos de los patrones GRASP como son: Experto, Alta Cohesión, Bajo Acoplamiento y Controlador. Sin embargo en la creación de las estructuras que conformarán el resultado de la investigación se decidió utilizar los siguientes:

- **Bajo Acoplamiento:** Para el funcionamiento del sistema y la reducción de la dependencia entre las clases, se crearon paquetes, donde cada cual se encargará de gestionar su propio negocio. Ejemplo de ello son los paquetes PTMBundle, ControlClaseBundle, GuardiaDocenteBundle, PerfilBundle.

³⁴ Del inglés: General Responsibility Assignment Software Patterns.

³⁵ Patrones que fueron creados por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides en 1995.

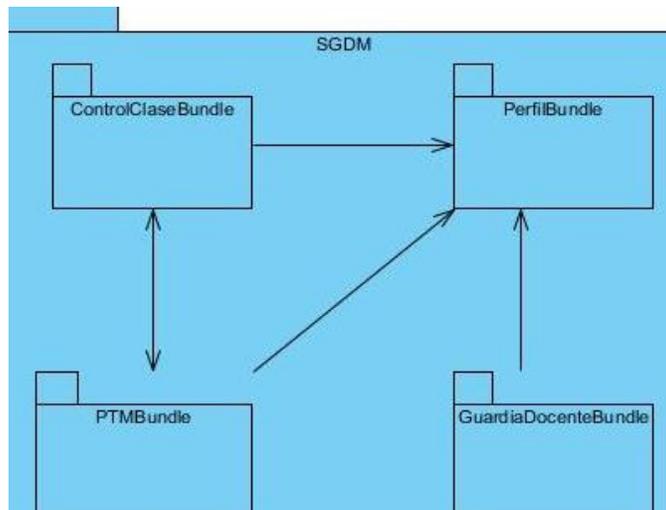


Figura 10 Representación de todos los paquetes existentes en el sistema.

- **Alta Cohesión:** En cada paquete existe una clase que contiene cantidad de funcionalidades y responsabilidades que son fundamentales para la ejecución del negocio, ya que tiene gran parte de la información de las clases que están en el paquete. Como ejemplo se evidencia: Paquete PTM en sus entidades la clase PlanTrabajoMetodologico.

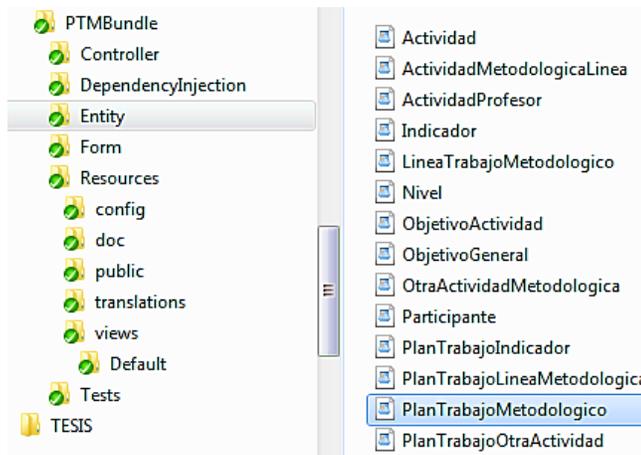


Figura 11 Aplicación del patrón Alta Cohesión en las entidades del caso de estudio.

- **Controlador:** Se requiere que una clase sea la encargada de las funcionalidades y responsabilidades que sean del negocio de ese paquete. Ejemplo de ello, es la clase principal PlanTrabajoMetodologico del paquete PTMBundle, porque se encarga de las funcionalidades y responsabilidades de su negocio. Ya que la misma puede modificar y devolver atributos, además de hacer operaciones de cálculos entre otras.

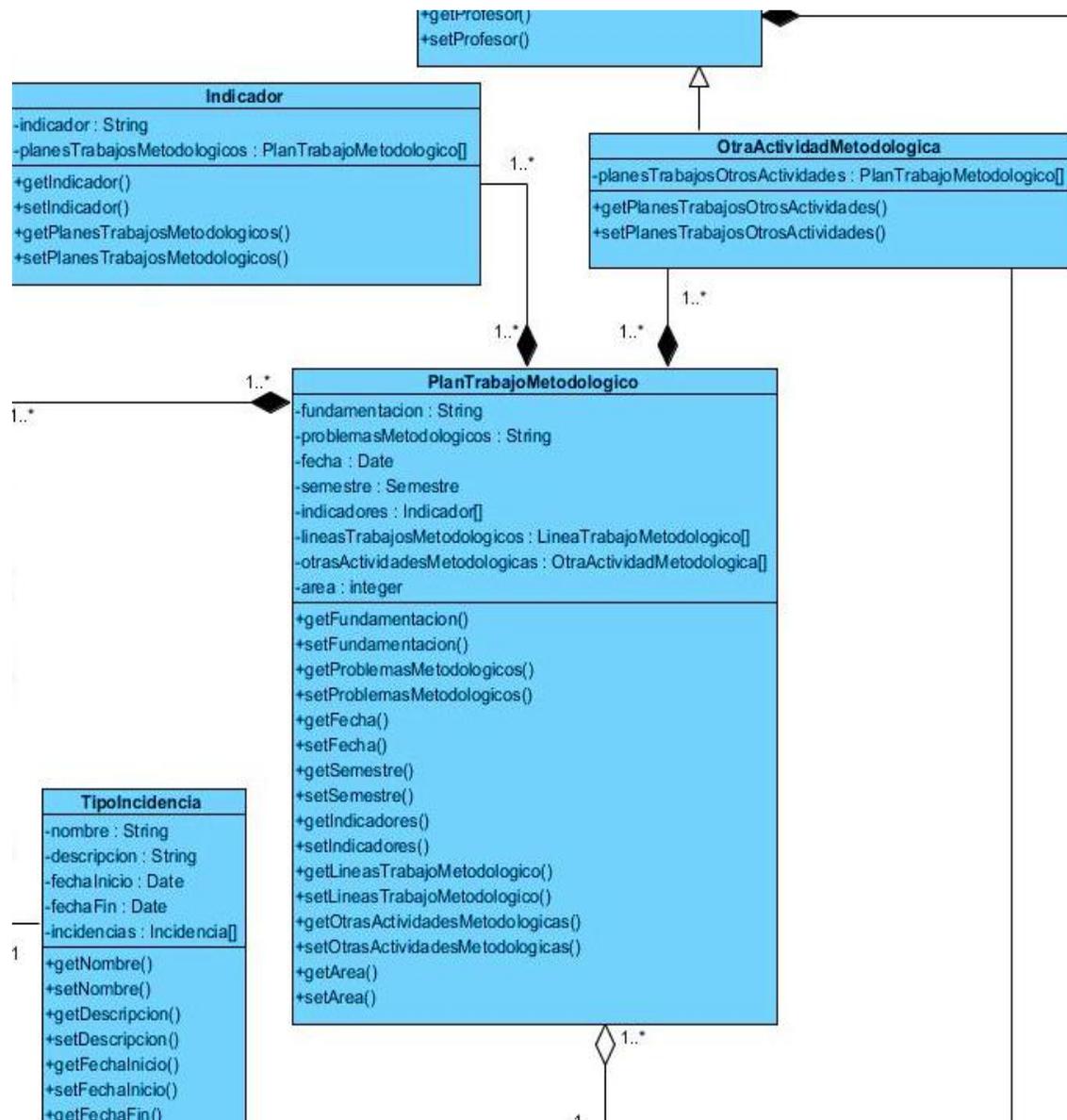


Figura 12 Aplicación del patrón Controlador en las clases del caso de estudio.

- Experto:** Al necesitar una clase que se encargue de la gestión de los datos de un profesor que se encuentran en el Subsistema de Gestión Integral de Personas (SGIP), se hace indispensable el uso de este patrón. Como ejemplo se tiene la clase Profesor que se especializa en la gestión de los datos de una persona, o sea es experta en la comunicación con el subsistema, para ello dentro de sus atributos se encuentra el CI que sería el identificador.

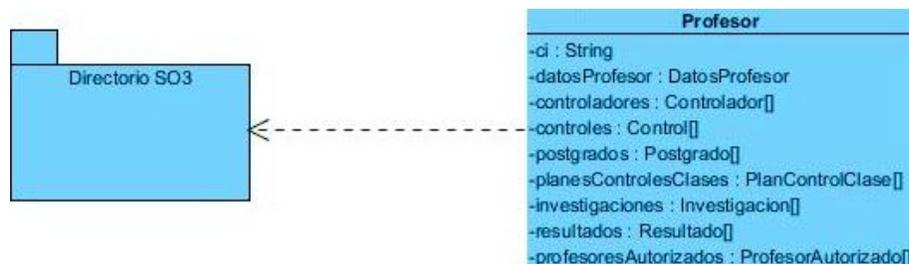


Figura 13 Aplicación del patrón Experto en la entidad Profesor.

Además, en este mismo ejemplo visual se evidencian los siguientes patrones GoF:

- **Proxy:** Este patrón proporciona un sustituto representante de otro objeto para controlar el acceso. Ejemplo de ello es la clase Profesor, ya que es un objeto representante de la clase persona porque mediante la misma, nuestro sistema puede tener los datos de una persona.
- **Mediador:** Se define que un objeto encapsula cómo interactuar un conjunto de objetos. Como ejemplo se tiene la clase Profesor que es mediador entre el Subsistema de Gestión Integral de Personas (SGIP) perteneciente a SO3 y el sistema a desarrollar en esta investigación.

En adición a los patrones GoF que emplea Symfony2: Factoría, Fachada, Composición, Constructor, Inyección de Dependencia, Decorador y Observador, que también son GoF, se aplicaron otros como es el caso de los siguientes:

- **Inyección de Dependencia:** Este patrón contribuye a la mantenibilidad y escalabilidad del sistema, ya que permite crear objetos de una clase, para inyectárselos a otras clases que dependen de la anterior. Ejemplo de ello es adicionar un controlador para realizar un control planificado, los controladores pertenecen a la clase Controlador, de esta clase se crea un objeto (controladorObj) y se le inyecta a la clase Control.

```

$controladorObj = new Controlador();
$controladorObj->setProfesor($profesor);
$control->addControlador($controladorObj);
  
```

Al tenerse definidos los patrones a emplear, la información proveniente del modelado y el prototipado, se pueden realizar los diagramas de clases del diseño. Específicamente el que se muestra a continuación es el perteneciente al caso de estudio.

2.9 Diagrama de Clases del Diseño

El diagrama está compuesto por la vista inicioPTM, en la cual se encuentra un enlace a la vista crearPTM, esta última es construida por la página servidora app.php. La página cliente crearPTM incluye las bibliotecas js del proyecto, además del archivo JavaScript crearPTM.js que contiene la lógica y las validaciones realizadas en el navegador del cliente. Crear PTM está compuesto por el formulario CrearPTMForm y a su vez este último está compuesto por otros formularios que manejan los restantes datos involucrados en el proceso. Después de ser enviado los datos del formulario, pasan al controlador frontal el que los redirecciona al DefaultController donde son procesados y validados.

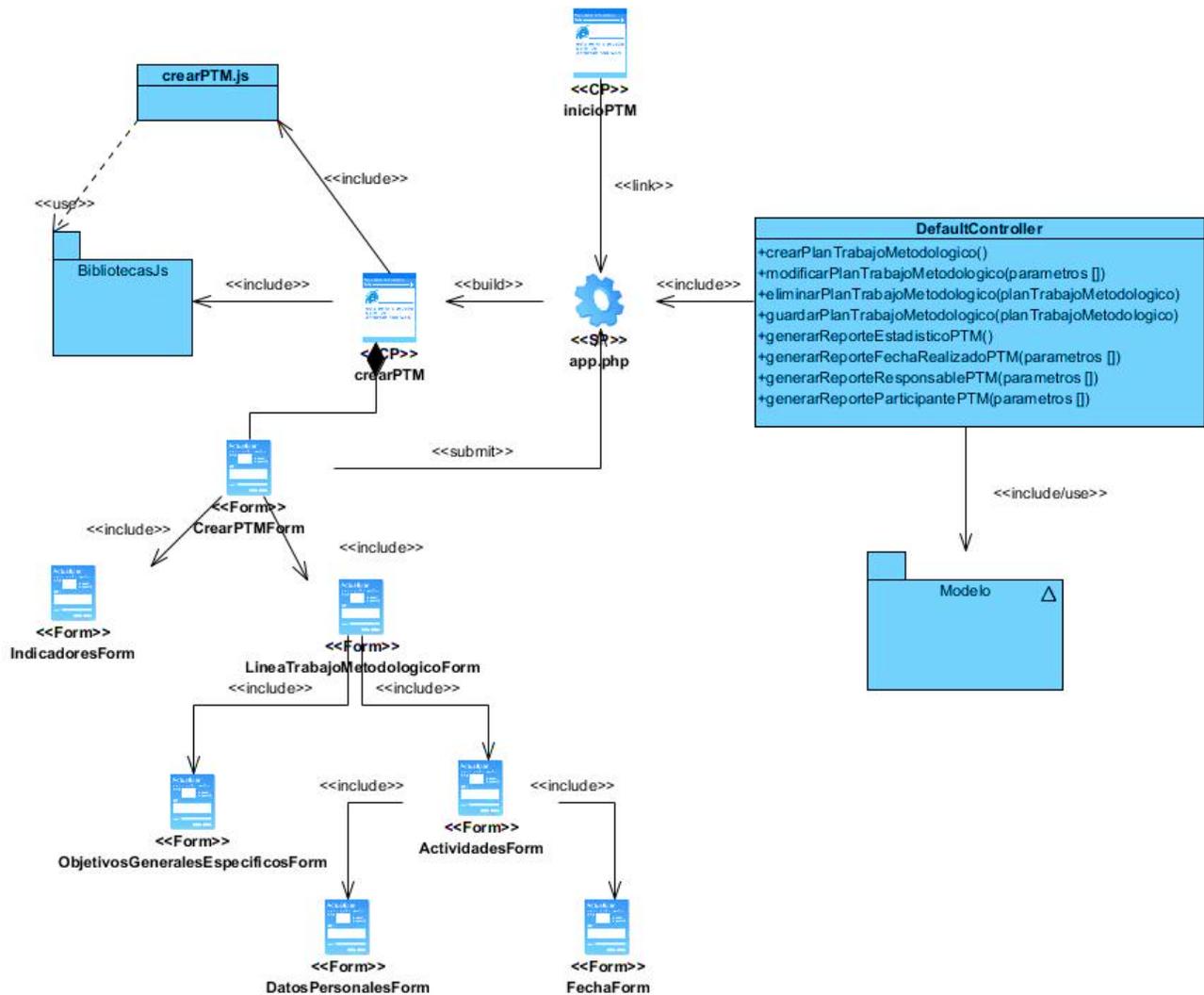


Figura 15 Diagrama de Clases del diseño: Crear Plan de trabajo metodológico.

CONCLUSIONES

Los conceptos más importantes del negocio quedaron recogidos en el mapa conceptual, de manera que la comprensión de todas las especificidades se simplificó. A partir del caso de estudio se mostraron los artefactos de la investigación y se describieron algunos de los patrones empleados en la propuesta. Además las tecnologías seleccionadas facilitaron toda la representación de la información aportando en el caso de los modelados y el prototipado una gran cantidad de información visual con uso de simbología e iconografía que simplificaban la comunicación cliente-equipo de desarrollo.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS

3.1 Introducción

En este capítulo se abarcará desde la implementación a la validación. Aquí se presentará la selección de los estándares de implementación, la validación de todos los artefactos que fueron generados hasta este momento que estén relacionados directamente con el negocio. Además de ejemplificar el uso de los patrones ya seleccionados en el capítulo anterior, definir las métricas de software que permitirán medir la implementación realizada para saber su grado de aceptación y la realización de pruebas para ver su efectividad.

3.2 Estándares de codificación

Teniéndose el caso de que el equipo de desarrollo de la presente investigación cuenta con dos estudiantes de 5to año y que una vez concluido el sistema y obtenida la aceptación del cliente, los resultados pasarán a ser parte de los paquetes que presenta la Plataforma Informativa SO3. La actual investigación se rige por la especificación del estándar de codificación adoptados por la Plataforma SO3, debido las características antes mencionadas y se puede encontrar en el expediente de proyecto. Estos estándares serán reglas a seguir a la hora de escribir código fuente para mantener un trabajo coordinado, de buena calidad y una misma forma de implementación que brinde la idea de ser una sola persona en todo el código (Microsoft, 2013).

Una vez concluida la implementación es preciso comenzar a validar toda la propuesta desde el inicio hasta el entregable que se obtiene: la aplicación.

3.3 Validación de los artefactos

Cada artefacto al ser confeccionado fue revisado por todo el equipo de desarrollo y los tutores de la investigación, para su posterior registro en el expediente de proyecto como guía futura. Debido al uso de la metodología SXP y la notación BPMN para el modelado del negocio, se obtuvo una mayor integración con el cliente, permitiendo la revisión y corrección de los modelados por su fácil entendimiento y la emisión de una carta de aceptación para cada uno de ellos. Como el cliente en cuestión es experto en el funcionamiento de cada proceso de acuerdo a los artículos del Reglamento Docente y Metodológico, Resolución No. 210/07, es el único que puede emitir una carta que valide toda la gestión del negocio hecha en el análisis de la propuesta. En el caso de los PIU, se presentaron a varios de los usuarios finales, así como

ejemplos del sistema una vez implementado, para recoger sus opiniones y modificarlos acorde a sus expectativas, facilitando una interfaz amigable y fácil de manejar sin tener que aprender.

3.3.1 Métricas de software

Durante el desarrollo de la solución se hizo necesario medir la calidad de los entregables y del producto para poder realizar acciones correctivas en tiempo y forma. Para ello se utilizaron una serie de métricas que permiten calcular variables cualitativas como la calidad, corrección y progreso de cada elemento.

Métrica para la calidad de la especificación de requisitos

Una vez revisados los modelos de procesos y sus descripciones, se hizo necesario evaluar la falta de ambigüedad, facilidad de comprensión y el grado de avance de la especificación de requisitos (Pressman, 2005). Llegando a identificar la cantidad exacta de requisitos en el desarrollo de esta investigación, se encuentran los siguientes datos:

Elemento de la fórmula: Nr = Nf + Nnf	Significado	Resultados obtenidos
Nf	número de requisitos funcionales	41
Nnf	número de requisitos no funcionales	7
Nr	cantidad de requisitos en una especificación	48

Tabla 8 Aplicación de la fórmula general de la métrica.

Donde se evidencia que para la fórmula especificada existe un total de 48 requisitos. En pos de determinar la falta de ambigüedad de los mismos, se hace uso de una métrica basada en la interpretación de la revisión de cada requisito por diferentes personas vinculadas al departamento de Ingeniería de Software de la Facultad 3, donde se llegó a revisar los requisitos del administrar plan, administrar planificación de control a clase, administrar planilla de control a clase y su generar reporte, además de administrar distribución y descripción de la guardia docente, adicionar y eliminar profesores al listado de la guardia docente, obteniéndose los siguientes resultados:

Elemento de la fórmula: Q1 = Nui / Nr	Significado	Resultados obtenidos
Nui	número de los requisitos que todos los revisores interpretaron de la misma manera	23

Nr	cantidad de requisitos en una especificación	41
Q1	valor entre 0 y 1	0.56

Tabla 9 Cálculo de la falta de ambigüedad en los requisitos de la investigación.

Evidenciándose que $Q = 0.56$, siendo un valor cercano a 1 por lo que la ambigüedad será menor. Sin embargo esta métrica no aporta de manera general el grado de avance de los requisitos, tanto funcionales como no funcionales. Dentro de los mismos se llegaron a validar todos los funcionales por el cliente y los no funcionales por el arquitecto de la Plataforma Informativa SO3 dentro de los que se encuentran: usabilidad, apariencia o interfaz externa, diseño e implementación, hardware, software y seguridad, aportando los siguientes datos:

Elemento de la fórmula: $Q3 = Nc / [Nc + Nnv]$	Significado	Resultados obtenidos
Nc	número de requisitos que se han validado como correctos	47
Nnv	número de requisitos que aún no se validan	1
Q3	grado de validación de los requisitos	0.98

Tabla 10 Grado de avance en los requisitos de la investigación.

Dando como resultado aproximadamente 0.98% de requisitos validados, siendo estos cercanos al valor 1, denotándose por ende un alto grado de avance de manera general, aunque resta verificar el soporte una vez desplegada la aplicación.

Métrica para el diseño: Tamaño Operacional de Clase (TOC)

En el diseño se aplicó además la métrica Tamaño Operacional de Clases (TOC) propuesta por Lorenz y Kidd (Pressman, 2005). Para su realización se contaron los atributos y operaciones (ya sean heredados o privados de la instancia) de una clase. Llegando a evaluar parámetros como la reutilización, responsabilidad y complejidad, determinando para cada parámetro los posibles problemas que puede acarrear valores altos del TOC. Para el entendimiento de la aplicación de esta métrica se tomó como ejemplo la clase Controlador del paquete ControlClaseBundle llegando a obtener los siguientes datos:

No	Clase	Operaciones	Atributos	Umbral	Tamaño
1	Controlador	5	3	8	Pequeño

Tabla 11 Clase del requisito Administrar Control a Clase aplicando la métrica.

Donde se puede apreciar que el valor obtenido, llamado umbral, se encuentra dentro de las siguientes clasificaciones en el TOC, específicamente menor que 20.

TOC	Umbral
Pequeño	Umbral \leq 20
Medio	Umbral $>$ 20 y Umbral \leq 30
Grande	Umbral $>$ 30

Tabla 12 Umbrales para la métrica TOC.

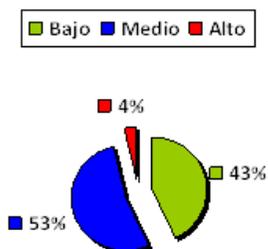
Si existen valores grandes de TOC éstos estarán demostrando que una clase puede tener demasiada responsabilidad, lo cual reduciría la reutilización de la clase y hará complicada la implementación. Ya obtenida la cantidad de operaciones, atributos y umbrales, se prosiguió a obtener el promedio de los umbrales y compararlos siguiendo los criterios siguientes:

Característica	Categoría	Criterio
Responsabilidad	Baja	Umbral \leq promedio.
	Media	Promedio $<$ Umbral \leq 2^* promedio
	Alta	Umbral $>$ 2^* promedio
Complejidad implementación	Baja	Umbral \leq Promedio
	Media	Promedio $<$ Umbral \leq 2^* promedio
	Alta	Umbral $>$ 2^* promedio
Reutilización	Baja	Umbral $>$ 2^* promedio
	Media	Promedio $<$ Umbral \leq 2^* promedio
	Alta	Umbral \leq Promedio

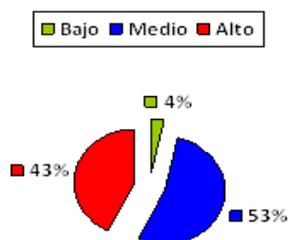
Tabla 13 Rango de valores para medir la afectación de los atributos de calidad (TOC).

Después de realizado todos los cálculos se obtiene para la clase tomada como ejemplo una responsabilidad y complejidad baja, mientras que se obtuvo una reutilización alta. Este mismo procedimiento se aplicó a un total de 28 clases, para un total de 161 atributos, 317 operaciones y un promedio de operaciones de 17,07 aproximadamente. De todas las clases analizadas se denotan los resultados en los siguientes gráficos:

Responsabilidad y Complejidad



Reutilización



Teniéndose como total:

- 1 Clase de tamaño Alto
- 15 Clases de tamaño Medio
- 12 Clases de tamaño Bajo

Teniéndose como total:

- 12 Clases de tamaño Alto
- 15 Clases de tamaño Medio
- 1 Clase de tamaño Bajo

Como se muestra en los gráficos anteriores, en el caso de la responsabilidad se tiene que existen muchas clases que tienen una responsabilidad baja o media debido a que los valores del TOC son relativamente bajos. Según los resultados gráficos se concluye que la responsabilidad, complejidad y reutilización de las clases es medio. Conllevando a no presentar complejidad en la implementación y las pruebas del sistema, además de que aumenta la reutilización en las clases.

Métrica para el diseño: Relaciones entre Clases (RC)

Para el funcionamiento del sistema cada clase se relaciona con otras para manipular su información mediante relaciones. La métrica Relaciones entre Clases está basada en el número de relaciones de uso de una clase con otra. Todo ello se realiza con el objetivo de medir atributos de calidad en la reutilización, acoplamiento³⁶, complejidad de mantenimiento y cantidad de pruebas³⁷ (Pressman, 2005).

Para aplicar esta métrica se hizo un Conteo de Relaciones de Uso (CRU) que posee cada una de las clases en el sistema. Luego se procedió a calcular el promedio de las mismas y con

³⁶ Grado de dependencia o interconexión de una clase o estructura de clases, con otras.

³⁷ Grado de esfuerzo para realizar las pruebas de calidad (Unidad) del producto (componente, módulo, clase, conjunto de clases, etc.) diseñado.

ambos valores según los criterios expuestos en la tabla siguiente (*Tabla 12*) se determinaron los valores de cada elemento ya anteriormente mencionado.

Atributos de calidad	Clasificación	Criterio
Acoplamiento	Ninguna	CRU = 0
	Baja	CRU = 1
	Media	CRU = 2
	Alta	CRU > 2
Complejidad de mantenimiento	Baja	CRU <= Promedio
	Media	Promedio < CRU <= 2* promedio
	Alta	CRU > 2* promedio
Reutilización	Baja	CRU > 2* promedio
	Media	Promedio < CRU <= 2* promedio
	Alta	CRU <= Promedio
Cantidad de pruebas	Baja	CRU <= Promedio
	Media	Promedio < CRU <= 2* promedio
	Alta	CRU > 2* promedio

Tabla 14 Rango de valores para medir la afectación de los atributos de calidad (RC).

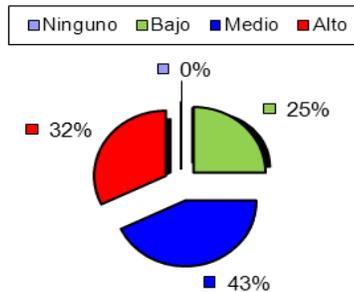
De esta manera se obtienen los valores de RC, si son elevados implica un aumento del acoplamiento de las clases, así como un aumento de la complejidad de mantenimiento de las mismas. En general, los valores de RC deben mantenerse tan bajos como sea razonable. Como ejemplo de la aplicación de esta métrica se ve a continuación en algunas clases del paquete ControlClaseBundle:

Clase	Cantidad de relaciones	Acoplamiento	Complejidad Mantenimiento	Reutilización	Cantidad de Pruebas
Control	4	Alto	Alta	Baja	Alta
Controlador	2	Medio	Baja	Alta	Baja
Evaluación	2	Medio	Baja	Alta	Baja

Tabla 15 Resultado de la aplicación de la métrica RC al paquete ControlClaseBundle.

Estos mismos pasos se aplicaron a un total de 28 clases del sistema llegando a obtener 72 Relaciones entre Clases de Uso (RCU), para un promedio de 2,57 aproximadamente. Llegando a confeccionar de sus resultados los siguientes gráficos:

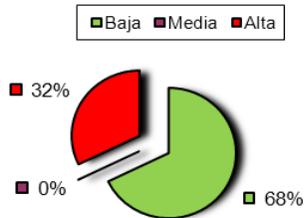
Acoplamiento



Teniéndose como total:

- 0 Clases con Ninguna
- 9 Clases de Alto
- 12 Clases de Medio
- 7 Clases de Bajo

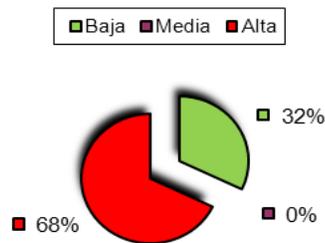
Complejidad de Mantenimiento y Cantidad de Pruebas



Teniéndose como total:

- 9 Clases de Alto
- 0 Clases de Medio
- 1 Clase de Bajo

Reutilización



Teniéndose como total:

- 19 Clases de Alto
- 0 Clases de Medio
- 9 Clase de Bajo

Como se muestra en los gráficos anteriores, los valores del RC son mínimos, por lo cual se concluye que la mitad de las clases existentes presentan un grado medio de dependencia entre clases y una gran reutilización de manera general. Mientras que existirá poca complejidad para brindar mantenimiento y requerirá de pocas pruebas de calidad del producto.

3.4 Pruebas

Las pruebas del software permiten verificar y validar la calidad del sistema a través de pruebas al diseño y la codificación, entre otras. A nivel general, la etapa de pruebas del software consiste en determinar los posibles errores que puedan existir, probar si el software hace lo que debe o si se comporta de otra manera. Para ello, se aplican las pruebas, métodos de prueba y técnicas necesarias para poder evaluar el sistema de acuerdo a los parámetros que se quieran

probar. Estas pruebas deben validar que el sistema realmente constituye una respuesta de calidad a las necesidades del cliente.

3.4.1 Pruebas del nivel de Unidad

Como parte del propósito de probar cuán acertada resultó la implementación de las funcionalidades del sistema, se decide aplicar del nivel de pruebas unitarias, el método de caja negra, que recibe su nombre ya que la evaluación funcional de los requisitos es transparente al usuario. Para esta prueba, se seleccionó y aplicó la técnica de partición equivalente, donde se obtuvieron los artefactos denominados: Casos de prueba. A través de este tipo de evaluación se puede demostrar que las funciones del software son operativas y que se obtiene el resultado esperado con la integridad de la información manipulada (Pressman, 2005).

Diseño de casos de prueba

A partir de la necesidad expresada por el cliente en función de obtener un sistema amigable y fácil de trabajar, se procedió a probar el medio de interacción del cliente, la interfaz de usuario. La cual a partir de un conjunto de elementos de entrada (válidos e inválidos) verificará si las salidas obtenidas son correctas o no, permitiendo identificar No conformidades al respecto. Por lo tanto, a mayor cantidad de elementos a probar, mayor probabilidad de encontrar problemas en el software y por lo tanto, será más confiable la calidad del mismo cuando sea liberado (Pressman, 2005). Para su documentación como ejemplo en la presente investigación, se seleccionó el requisito Crear Planificación de Control a Clase el cual consiste en:

Descripción general	Condiciones de ejecución
El requisito Crear Planificación de Control a Clase permitirá crear un control a determinado profesor. Manipulando información como el profesor que se va a controlar, su categoría docente y asignatura, los controladores, el rango de fecha a ser controlado y la semana docente.	<ul style="list-style-type: none">• El sistema debe estar disponible.• Se debe identificar y autenticar ante el sistema y además tener los permisos para ejecutar esta acción.• Se debe seleccionar la opción Plan de Control a Clase para obtener un listado de todos los planes y seleccionar el que se desee para ver, crear, modificar y eliminar sus controles.

Tabla 3 Descripción general y condiciones de ejecución del requisito.

Como se mencionaba anteriormente existen un conjunto de variables que permiten el funcionamiento del sistema, de las cuales se tiene la siguiente descripción.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1.	Profesor	Campo de texto	No	Cadena, solo admite letras.
2.	Fecha de Inicio	Tipo fecha	No	Solo admite fecha en formato yyyy-mm-dd. La fecha debe estar en el semestre del plan.
3.	Fecha Fin	Tipo fecha	Si	Solo admite fecha en formato yyyy-mm-dd. La fecha debe estar en el semestre del plan e igual o mayor (rango una semana) que la fecha inicio.
4.	Controladores	Campo de texto	No	Cadena, solo admite letras.
5.	Estado	Lista desplegable	No	Cadena, solo admite letras.
6.	Semana Docente	Lista desplegable	No	Cadena, solo admite letras.

Tabla 4 Descripción de las variables del requisito Crear Planificación de Control a Clase.

En aras de determinar los posibles errores, se planificaron juegos de datos para probar los siguientes escenarios:

Escenario	Descripción	Profesor	Fecha de inicio del control	Fecha final del control	Controladores	Estado del control	Semana docente	Respuesta del sistema
EC 1.1 Crear la planificación correctamente	Al insertar los datos requeridos se crea la planificación.	V Rosalina Ibarra Gonzalez	V 10/09/2013	V 15/09/2013	V Dariela Elvira Espinosa Leyva	V Planificado	V 3	El sistema guarda correctamente los datos.
EC 1.2 Campos Incompletos.	No se crea la planificación porque quedaron campos vacíos.	I (vacío)	V 10/09/2013	V 15/09/2013	V Dariela Elvira Espinosa Leyva	V Planificado	V 3	El sistema muestra un mensaje de aviso informando que hay campos incompletos.
		V Rosalina Ibarra Gonzalez	I (vacío)	V 15/09/2013	V Dariela Elvira Espinosa Leyva	V Planificado	V 3	

Tabla 5 Ejemplo de datos de prueba del requisito Crear Planificación de Control a Clase.

Con la finalidad de tener mayor certidumbre sobre la calidad y funcionalidad del software desarrollado, se realizaron dos iteraciones de revisiones. Esta técnica permitió revelar un conjunto de errores que de otro modo, requerirían la ejecución de muchos casos de prueba. En la primera iteración de pruebas se detectaron 21 no conformidades, de ellas 13 fueron significativas, lo que representa un 61,9% del total. A continuación se muestran en el gráfico los resultados obtenidos.

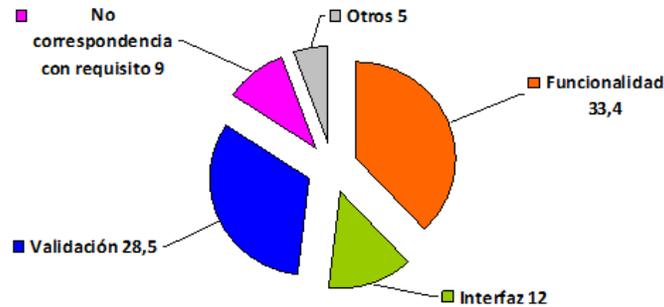


Figura 16 Resultados de la primera iteración de la prueba.

Las no conformidades referentes a funcionalidades y validación representan en mayor medida el nivel de afectación en la aplicación, no siendo así en las restantes. Todas las no conformidades fueron resueltas, sin embargo se hizo necesario comprobar la veracidad de dichas soluciones, dando paso a la segunda iteración. Durante la misma, se llegaron a detectar 9 no conformidades. De ellas 3 fueron significativas, lo que representa un 33% del total. A continuación se muestran los resultados.

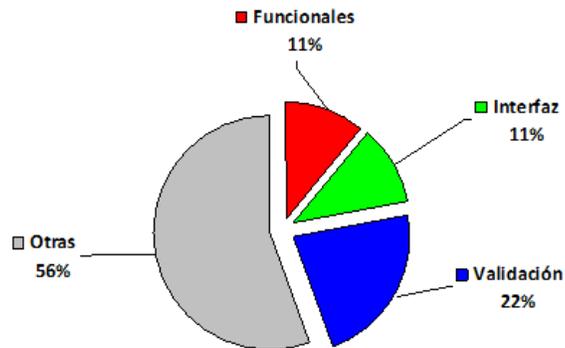
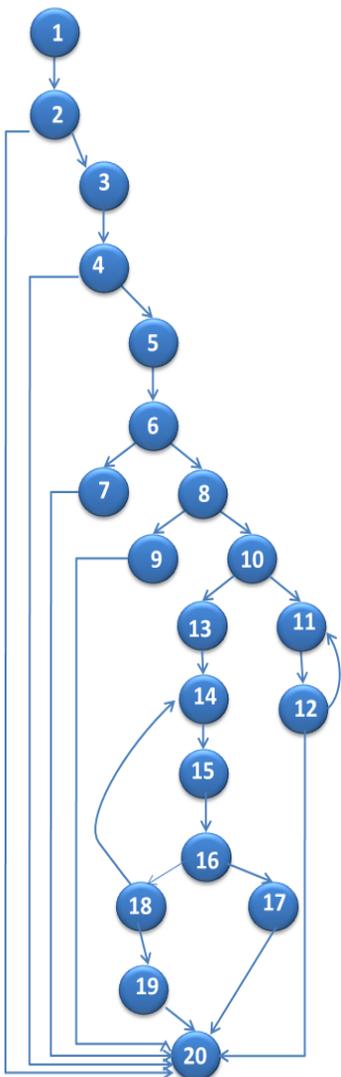


Figura 17 Resultados de la segunda iteración de la prueba.

Como se evidencia todos los valores fueron disminuyendo considerablemente, llegando a obtener entre las funcionalidades y las validaciones menores valores con respecto a los de la primera iteración. Sin embargo, se obtuvieron errores en cuanto a los elementos de la interfaz y

faltas de ortografía, mientras que se mantuvo una correspondencia con los requisitos. Por los resultados antes expuestos se pudo apreciar que durante la segunda iteración de pruebas, disminuyeron considerablemente el número de no conformidades detectadas, demostrando con ello el aumento de la calidad del software y una mejor correspondencia de los procesos descritos con su implementación. Conllevando a una mayor satisfacción de las necesidades del cliente. Luego de completar las pruebas anteriores y la finalidad de saber si el código implementado se encuentra en perfecto estado y que al menos es ejecutado una vez, se seleccionó del nivel de pruebas unitarias, el método de caja blanca, para garantizar que se ejerciten por lo menos una vez todos los caminos independientes del módulo, así como la ejecución de todos los bucles en sus límites operacionales y todas las decisiones lógicas en las vertientes verdaderas y falsas (Pressman, 2005).

Técnica del camino básico



La técnica del camino básico, es una de las pruebas que pueden hacerse al código, la cual muestra la complejidad lógica de un diseño y permite garantizar que los casos de prueba obtenidos se ejecutan por lo menos una vez (cada sentencia del programa). Para ello hace uso de la notación gráfica de flujo para cada estructura, mediante nodos (círculos) y enlaces (flechas) (Pressman, 2005). La muestra de los resultados de esta técnica, se evidencia en la funcionalidad `crearControlClase()`, el cual consiste en planificar una evaluación a la clase de cierto profesor. Una vez definida la funcionalidad a examinar, se procedió a realizar su prueba, donde se evidenció en el código el uso de estructuras `if()` y `for()`, ofreciéndole la forma actual al grafo representado a la izquierda. Permitiendo entonces calcular su **complejidad ciclomática** con las siguientes fórmulas:

- $V(G)=9$
- $V(G)=A-N+2=27-20+2=9$
- $V(G)=P+1=8+1=9$

A partir de los resultados se tiene que 9 es el límite máximo de rutas independientes, determinándose por tanto los siguientes caminos:

Camino básico #1: 1-2-20

Camino básico #2: 1-2-3-4-20

Camino básico #3: 1-2-3-4-5-6-7-20

Camino básico #4: 1-2-3-4-5-6-8-9-20

Camino básico #5: 1-2-3-4-5-6-8-10-11-12-20

Camino básico #6: 1-2-3-4-5-6-8-10-11-12-11-12-20

Camino básico #7: 1-2-3-4-5-6-8-10-13-14-15-16-17-20

Camino básico #8: 1-2-3-4-5-6-8-10-13-14-15-16-18-19-20

Camino básico #9: 1-2-3-4-5-6-8-10-13-14-15-16-18-14-15-16-18-19-20

Al existir 9 rutas como máximo, se tiene entonces que el límite de pruebas a ejecutar donde se prueben todos los caminos es de 9. Para demostrar la aplicación de las pruebas a los caminos, se tiene como ejemplo el siguiente caso de prueba del requisito Crear Planificación de Control a Clase específicamente en el camino básico # 9.

Caso de prueba: Camino básico # 9	
Entrada	Como entrada se debe tener el plan al que corresponde el control que se desea planificar. Se necesita que la petición se haya realizado por el método POST y que los datos del formulario sean válidos. Se debe haber seleccionado el botón Guardar y que no existan errores en los datos enviados de acuerdo a las reglas del negocio.
Resultados esperados	Los datos del control planificado se salvan en la base de datos y se redirecciona hacia la ruta <code>listarControlesClases()</code> .
Condiciones	<pre>\$planControlClase!=null \$request->getMethod() = 'POST' \$form->isValid()==true count(\$form->getErrors())=0</pre>

Tabla 6 Caso de prueba de caja blanca para el camino básico # 9.

En una primera iteración de estas pruebas, se detectaron un total de 21 errores, para luego dar paso a la segunda iteración. De dichas pruebas se obtuvieron resultados satisfactorios, llegando a demostrar que los caminos se ejecutaban al menos una vez en el sistema, que los métodos funcionaban correctamente, así como las diferentes estructuras utilizadas.

3.4.2 Pruebas del nivel de Sistema

Anteriormente se realizaron pruebas al código y al diseño resultando la verificación del funcionamiento y de la calidad del código. Sin embargo, aun no es suficiente, es preciso aplicarle a la solución pruebas del nivel de sistema conocidas como Pruebas de Resistencia y Rendimiento, para validar la capacidad del sistema para enfrentar situaciones anormales.

Prueba de Resistencia y Rendimiento

Este tipo de pruebas por lo general son realizadas por programas encargados de simular condiciones extremas para el software a probar, aunque también pueden ser realizadas manualmente. Para poder cumplir con este requisito, se requirió de una herramienta con las características necesarias para ejecutar las pruebas y apoyar todo el proceso (Almenares, 2009).

Las necesidades indicaban una herramienta para Aplicación Web, y por permitir la ejecución de pruebas distribuidas entre distintos ordenadores, activar o desactivar una parte de la prueba y realizar casos de prueba a través de una navegación de usuario, se decidió utilizar la herramienta **JMeter**. De acuerdo a las funcionalidades que presenta, la cantidad de resultados de las pruebas que permite obtener y ser un software libre y gratis, se convirtió en la seleccionada para todo este proceso.

Como parte de las pruebas se requirió priorizar algún requisito fundamental de los procesos Plan de trabajo metodológico, Control a Clases y Guardia Docente, además de algunos menos prioritarios. Para la obtención de los resultados, se utilizó una estación de trabajo con las siguientes características: Memoria de 1 GB RAM DDR2, Microprocesador Intel™ Dual Core™ 2.2 GHz, Sistema operativo Linux Mint 12, Servidor de aplicaciones Apache 2.2, Versión de JRE21 para JMeter: 7. Las primeras pruebas se iniciaron con una muestra de 10 peticiones, las cuales arrojaron como resultados un 2.28% de error en las peticiones. A continuación se muestran tablas y gráficas que complementan los datos obtenidos en las pruebas.

No	Funcionalidad	Peticiones ³⁸	Promedio ³⁹ (ms)	Media ⁴⁰ (ms)	Mínima ⁴¹ (ms)	Máximo ⁴² (ms)
----	---------------	--------------------------	--------------------------------	-----------------------------	------------------------------	------------------------------

³⁸ Cantidad de peticiones concurrentes realizadas por el software.

³⁹ Promedio de tiempo de respuesta de las peticiones realizadas expresado en milisegundos.

⁴⁰ Media del tiempo de respuesta de las peticiones realizadas expresada en milisegundos.

1.	Crear Plan de Control a Clase	10	4344	4542	3507	4783
2.	Listar Plan de Control a Clase	10	2405	2439	1922	2977
3.	Listar Planificación de Control a Clase	10	3455	3506	3085	3946
4.	Selección de profesores	10	4683	4682	4212	5264
5.	Listar Distribución de la Guardia Docente	10	4221	4320	3481	4928
6.	Crear Plan de trabajo metodológico	10	3554	3752	3022	3915
7.	Crear Distribución Guardia Docente	10	3610	3604	2949	4303
	TOTAL	70	3753	3796	1922	5264

Tabla 7 Resultados de las pruebas de carga y estrés con 10 peticiones.

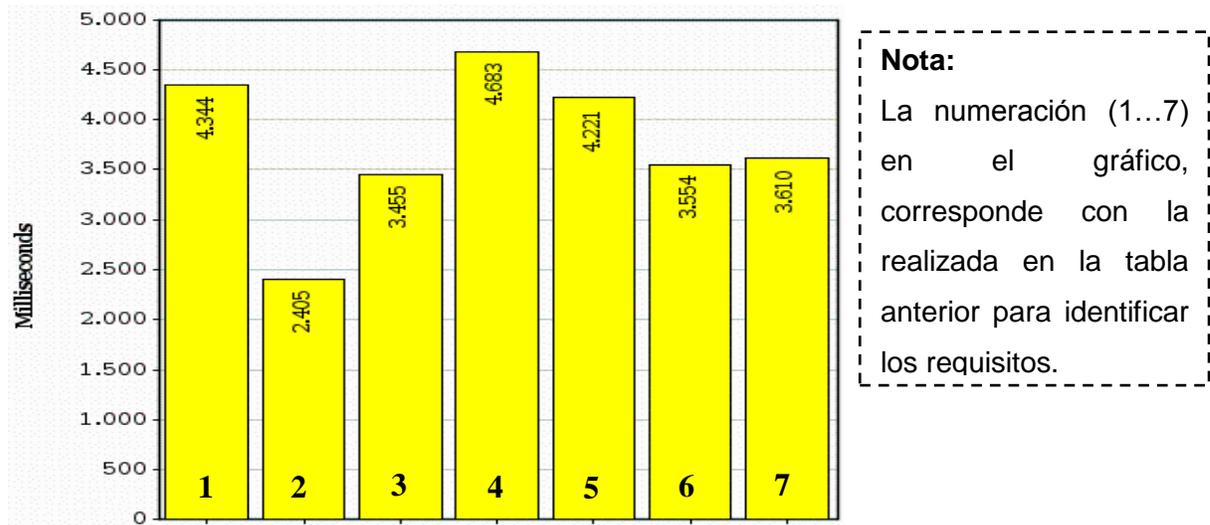


Figura 18 Resultados de la media de las pruebas de carga y estrés con 10 peticiones.

La realización de las pruebas, arrojó como resultado la existencia de tres funcionalidades (Selección de profesores, Crear Plan de Control a Clase y Listar Distribución de la Guardia Docente) cuyo tiempo de respuesta supera la de otras funcionalidades. Sin embargo estas se encuentran dentro de los límites aceptables y serán solicitados por un usuario o pocos respectivamente, en determinados momentos del curso. Además de que el flujo que siguen actualmente, es un requisito del cliente por lo que se determina mantener el procedimiento

⁴¹ Menor tiempo de respuesta expresado en milisegundos registrado por funcionalidad.

⁴² Mayor tiempo de respuesta expresado en milisegundos registrado por funcionalidad.

actual de la funcionalidad, pero optimizar las consultas a la base de datos y el procesamiento en el lado del servidor.

Al efectuarse una segunda prueba, luego de haber corregido los errores detectados en la primera iteración, se obtuvo como resultado un 100% de pruebas satisfactorias, con tiempo de respuesta promedio de 2.17 segundos en la estación de trabajo antes mencionada, el cual se puede calificar de satisfactorio. Teniendo en cuenta los resultados antes mencionados, se considera que en un entorno de producción con prestaciones iguales o superiores a las de la estación de trabajo, la respuesta promedio debe ser la misma o mejorada considerablemente. Se evidencia por tanto, una optimización realizada en algunas funcionalidades y el correcto funcionamiento en general de la aplicación.

Culminando toda la documentación se requirió tener la certidumbre de que todos los elementos tenían la calidad y estructura requerida, para ello se prosiguió a realizar la validación de los mismos.

CONCLUSIONES

Se establecieron los estándares de codificación permitiendo la programación independiente de cada uno de los paquetes. También se realizaron pruebas tanto al diseño como al código para evaluar los resultados obtenidos, llegando a tener respuestas favorables en cada caso. Se establecieron y aplicaron métricas para validar los artefactos y el diseño, generando un total de 0.98% de requisitos validados y un diseño donde la mitad de las clases existentes presentan un grado medio de dependencia entre ellas y una gran reutilización de manera general. Se obtuvieron además cartas de aceptación firmadas por el cliente, en lo concerniente a la aprobación de los artefactos del modelado, la descripción del negocio y sus reglas, así como los prototipos que darían lugar a las vistas del sistema.

CONCLUSIONES

Al finalizar la presente investigación se logró cumplir de manera satisfactoria los objetivos planteados al inicio de la misma, obteniéndose como resultado directo un sistema de gestión docente metodológico, que permite gestionar los procesos Control a Clases, Plan de trabajo metodológico y Guardia Docente, pertenecientes al núcleo de la Plataforma Informativa SO3.

Para el desarrollo de dicha solución fue necesaria la realización de un estudio profundo del estado del arte de las principales soluciones informáticas existentes para la gestión docente metodológica, la cual permitió establecer diferencias, fortalezas, debilidades así como puntos en común entre las mismas. Sin embargo, ninguno de los sistemas analizados cumplió con las especificaciones definidas, conllevando a realizar una selección de las herramientas, tecnologías y metodologías a utilizar para el desarrollo de la solución informática.

Posteriormente fueron identificadas las principales funcionalidades de la solución teniendo en cuenta los procesos antes mencionados. En aras de construir funcionalidades que cumplieran con las expectativas, se analizaron y seleccionaron los patrones de diseño a utilizar en la solución planteada y se diseñaron los diagramas establecidos en el flujo de implementación que dieron paso a la programación.

Concurrentemente a la implementación, se puso en prueba el correcto funcionamiento del sistema mediante la realización de pruebas a las funcionalidades y al código, además del rendimiento y resistencia del sistema. Llegándose a chequear no solo el cumplimiento de los requisitos planteados por el cliente, sino también el comportamiento de la aplicación bajo altos niveles de carga y estrés. Finalizando la validación, mediante el análisis de los resultados obtenidos en las pruebas y la aplicación de métricas, obteniéndose algunos resultados defectuosos, que luego de cada iteración se erradicaban para obtener valores satisfactorios.

La solución desarrollada aportará una mayor eficiencia y confiabilidad en la gestión docente metodológica de la Facultad 3, con información centralizada y disponible. Contribuyendo no sólo a la toma de decisiones por parte de los involucrados en el negocio, sino también a una eficiente gestión de la información.

RECOMENDACIONES

Los objetivos trazados en el presente trabajo de diploma fueron alcanzados satisfactoriamente, sin embargo se recomienda:

- Desarrollar nuevas funcionalidades que permitan la gestión de los datos de postgrado, ciencia y técnica de los profesores.
- Desarrollar funcionalidades que permitan el funcionamiento y gestión conjunta de los datos de los departamentos y asignaturas del sistema MOIRA con los de la Plataforma Informativa SO3.
- Implementar nuevos reportes en la medida de las necesidades.
- Definir un plan de capacitación para los usuarios que van a administrar la información por las diferentes áreas de la facultad.

BIBLIOGRAFÍA

1. **ADOSSIS, S.A. 1985-2012.** KRONOWIN-KGrupos Milenio-6. Refusiones para el Generador de Horarios y Generador de Grupos Milenio-6. [En línea] Noviembre de 1985-2012. [Citado el: 20 de noviembre de 2012.] <http://www.adossis.es/REFUGHM6.htm>.
2. **Agile Alliance. 2013.** Agile Alliance. [En línea] 2013. <http://www.agilealliance.org/>.
3. **Agile Spain. 2012.** Agile Spain. [En línea] 2012. www.agile-spain.com.
4. **Alvarez Hernández , Dailé y Martínez Vazquez, Elizeth . 2009.** *Sistema de Gestión para la Planificación Docente en la Facultad 5.* La Habana : s.n., 2009.
5. **Atego. 2010-2013.** Atego. [En línea] 2010-2013. <http://www.atego.com/community/standards/uml/>.
6. **Bahit, Eugenia. 2009.** *El paradigma de la Programación Orientada a Objetos en PHP y el patrón de arquitectura de Software MVC.* 2009.
7. **Batista, Yalice Gómez. 2013.** *Vicedecano de Formación.* La Habana, 25 de Marzo de 2013.
8. **Beacham, Nigel. 2012.** *Catalogue of GRASP design patterns.* 2012.
9. **Canal, Carlos. 2005.** *Arquitectura, marcos de trabajo y patrones.* 2005.
10. **Canós, José H., Letelier, Patricio y Penades, Maria Carmen. 2009.** *Métodologías Ágiles en el Desarrollo de Software.* Valencia : s.n., 2009.
11. **Castillo González, Lic. José Manuel , y otros. 2009-2010.** *La observación a clases: Vía fundamental para la superación pedagógica de los profesores.* Cuba : s.n., 2009-2010.
12. **Castledine, Earle y Sharkie, Craig. 2010.** *jQuery: Novice to Ninja by Earle Castledine and Craig Sharkie.* 2010.
13. **Colegium.com. 2012.** Colegium.com. [En línea] 2012. [Citado el: 20 de noviembre de 2012.] <http://www.colegium.com>.
14. **CristaLab. 2006.** CristaLab. [En línea] 17 de agosto de 2006. [Citado el: 7 de mayo de 2013.] <http://www.cristalab.com/tutoriales/reglas-de-codificacion-y-lineamientos-de-codigo-php-c190/>.
15. **Dr. Telot González, Julio, Lic. Calderon Delgado, Yanovski y Lic. Noda Rodríguez , Jorge. 2001.** *GESTACAD: Un sistema para la gestión académica universitaria.* Cuba, Matanzas : s.n., 2001.

16. **Ecured. 2010.** Ecured. [En línea] 14 de diciembre de 2010. [Citado el: 13 de junio de 2013.] <http://www.ecured.cu/index.php/Twig>.
17. **Erguiluz, Javier. 2012.** *Desarrollo web ágil con symfony2*. 2012.
18. **Folch, Patricia. 2012.** Documentacion Digital. [En línea] 19 de junio de 2012. <http://documentaciondigitalupf.wordpress.com/arquitectura-de-la-informacion/prototipado/axure/>.
19. **Gamma, Erich, y otros. 1995.** *Design Patterns: Elements of Reusable Object Oriented Software*. 1995.
20. **José Escalona, María y Koch, Nora. 2002.** *Ingeniería de Requisitos en Aplicaciones para la Web. Un estudio compartido*. España, Alemania : s.n., 2002.
21. **Juristo, Natalia, Moreno, Ana M. y Vegas, Sira. 2005.** *Técnicas de Evaluación de Software*. 2005.
22. **Katzke, Karl. 2013 .** [En línea] 2013 . <http://www.karlkatzke.com/php-symfony-vs-zend/>.
23. **Larman, Craig. 1999.** *UML y Patrones. Introducción al Análisis y Diseño Orientado a Objetos*. 1999.
24. **Leyet Fernández, Osmar, Rodríguez Lorenzo, losmel. 2008.** *Desarrollo de una herramienta generadora de ficheros de mapeo para la persistencia de esquemas de objetos relacionales basada en NHibernate*. Ciudad de la Habana : s.n., 2008. Tesis.
25. **López, Patricia y Ruiz, Francisco.** *Lenguaje Unificado de Modelado - UML*. Universidad Cantabria – Facultad de Ciencias : s.n.
26. **MASTERMAGAZINE. 2007-2012.** MASTERMAGAZINE. [En línea] 2007-2012. <http://www.mastermagazine.info/termino/5560.php>.
27. **Mehdi Achour, Friedhelm Betz. 1997-2013.** MANUAL DE PHP . [En línea] 1997-2013. <http://php.net/manual/es/preface.php>.
28. **Microsoft. 2013.** Microsoft. [En línea] Microsoft, 2013. [Citado el: 8 de mayo de 2013.] <http://msdn.microsoft.com/es-es/library/aa291593%28v=vs.71%29.aspx>.
29. **—. 2013.** MSDN. [En línea] Microsoft, 2013. [Citado el: 7 de mayo de 2013.] <http://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx>.
30. **Ministro de Educación Superior. 2007.** *Reglamento Docente y Metodológico. Resolución 210/ 07*. Cuba : s.n., 2007.
31. **Object Management Group, Inc. 2011.** Business Process Model and Notation (BPMN). [En línea] enero de 2011. <http://www.bpmn.org/>.

32. —. **2012.** OMG we set the standard. [En línea] 12 de abril de 2012.
http://www.omg.org/gettingstarted/what_is_uml.htm.
33. **OpenWebCMS. 2013.** OpenWebCMS. [En línea] 2013. [Citado el: 20 de junio de 2013.]
<http://openwebcms.es>.
34. **Oracle Corporation. 2012.** Netbeans. [En línea] 2012.
<http://netbeans.org/features/index.html>.
35. —. **2012.** Netbeans. [En línea] 2012.
<http://netbeans.org/community/releases/72/index.html>.
36. **Osorio, Luis Bernardo Duque. 2008.** *Procesos. Guía Didáctica Y Módulo.* Colombia : s.n., 2008.
37. **Peñalver, G., Meneses, A. y García, S. 2010.** *SXP, Metodología ágil para el desarrollo de software.* Cuba : s.n., 2010.
38. **Peñalver, G., Meneses, A., & García, S. 2010.** *SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE.* Cuba : s.n., 2010.
39. **Potencier, Fabien. 2007-2012.** Fabien Potencier. [En línea] 2007-2012.
<http://fabien.potencier.org/article/65/why-symfony> .
40. **Potencier, Fabien y Zaninotto, Francois. 2008.** *Symfony la guía definitiva.* 2008.
41. **Potencier, Fabien y Zaninotto, François. 2008.** *Symfony la guía definitiva.* 2008.
42. **Pressman, Roger S. 1998.** *Ingeniería de software un enfoque práctico.* 1998. 84-481-3214-9.
43. —. **2010.** *Ingeniería del software.* s.l. : McGraw-Hill, 2010. 7.
44. —. **2005.** *Ingeniería del Software. Un enfoque práctico.* Nueva York : Mc Graw Hil, 2005.
45. **Reyes, Zenel. 2011.** *Plataforma tecnológica de la Facultad 3.* La Habana : s.n., 2011.
46. **Samada, Lisandra Isabel Leyva. 2009.** *Flujo de Investigación para la Metodología.* La Habana : s.n., 2009.
47. **Scrum methodology. 2009.** Scrum methodology. [En línea] 2009.
<http://www.scrummethodology.org/scrum-advantages.html>.
48. **Scrum.org. 2013 .** Scrum . [En línea] 2013 . <http://www.scrum.org/>.
49. **SearchSQLServer.com. 2005-2013.** SearchSQLServer. [En línea] 2005-2013.
<http://searchsqlserver.techtarget.com/definition/database-management-system>.
50. **Sensio Labs. 2010-2012.** Sensio Labs. [En línea] 2010-2012. [Citado el: 5 de mayo de 2013.] <http://twig.sensiolabs.org/>.
51. **Sklar, David y Trachtenberg, Adam. 2003 .** PHP_Cookbook. [En línea] 2003 .

52. **TELLEZ, LINDA LUNA. 2012.** Metodología Ágil vs Metodología Tradicional. [En línea] 2012. [Citado el: 14 de 03 de 2013.] <http://es.scribd.com/doc/91676941/Metodologias-agiles-vs-tradicionales>.
53. **The PHP Group. 2001-2013.** PHP. [En línea] 2001-2013. <http://www.php.net/>.
54. **The JQuery Project. 2010 .** The JQuery Project. [En línea] 2010 . http://docs.jquery.com/Browser_Compatibility jquery write less. Do more.
55. **The PHP Group. 2001-2013.** PHP. [En línea] 2001-2013. <http://php.net/manual/es/preface.php>.
56. **Three Rivers Systems. 2010.** *California's William Jessup University selects CAMS Enterprise™ as its academic ERP to manage its entire student lifecycle.* California : s.n., 2010.
57. **UML. 2003-2012.** UML Forum. Everythings related to UML . [En línea] 2003-2012. <http://www.umlforum.com/faq/>.
58. **Verdecia Álvarez, Yoandrys y Hurtado Portal, Yoisbel. 2007.** *Sistema de Gestión de Informatización de la Facultad 8. Módulo para la Gestión de Planificación Docente.* La Habana : s.n., 2007.
59. **Visual Paradigm. 1999-2013.** Visual Paradigm. [En línea] 1999-2013. <http://www.visual-paradigm.com/product/vpuml/provides/bpmodeling.jsp..>
60. **—. 1999-2013.** Visual Paradigm. [En línea] 1999-2013. <http://www.visual-paradigm.com/product/vpuml/editions/community.jsp>.