



CENTRO DE INFORMATIZACIÓN UNIVERSITARIA
FACULTAD 1

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

Título: Solución informática para la gestión de información en eventos en
el Sistema de Gestión Universitaria

Autores:

Ernesto Julio Domínguez Rodríguez

María Caridad Escalona Mesa

Tutores:

Ing. Alelí Sánchez Méndez

Ing. Yanio García Vidal

Ing. José Alejandro García Calderón

La Habana
19 de Junio del 2012

Declaración de autoría

Declaro que soy el único autor de este trabajo y autorizo al Centro de Informatización Universitaria de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Ernesto Julio Domínguez Rodríguez

María Caridad Escalona Mesa

Firma del autor

Firma del autor

Ing. Alelí Sánchez Méndez

Ing. Yanio García Vidal

Firma del tutor

Firma del tutor

Ing. José Alejandro García Calderón

Firma del tutor



Dedicatoria

Mary

A mis padres, por haberme dedicado sus vidas primero.

Ernesto J

Ante todo me lo dedico a mi mismo.

A mis padres que me han apoyado en todo momento. Ellos lo son todo para mí, y espero que se sientan orgullosos de su niño.

A mi hermana que no está a mi lado en estos momentos pero lo está en mi corazón.

A mi novia que ha sido todo para mí en la UCI.

A mis abuelos Julio y Celia que no están ya con nosotros pero fueron la guía de los primeros pasos de mi niñez. También mi abuelo Miro que tampoco está hoy en vida, por brindarme tantos conocimientos en el deporte y a mi abuela Clarita por darme tanto amor y cariño.

A mi familia y amigos.



Agradecimiento

Mary

Gracias a mi familia, que se ha sacrificado conmigo y nunca me ha cortado las alas. Mis amistades que me han aguantado tanto, y me han enseñado a no renunciar, ellos saben quiénes son.

A mis padrinos, que han esperado el momento durante 5 años. A mi esposo, que no me ha dejado caer. A mis suegros, que han sido los mejores. A las amistades de mi pueblecillo. A los que he molestado todo el tiempo.

A mi tutora, que se sacrificó tanto para llegar a este momento.

Ernesto J

Le agradezco a mis padres por haberme dado la vida, por haber sido mi inspiración, por ser la voz de la experiencia que ha guiado todo mi camino.

A mi hermana y mi cuñado que no están aquí conmigo pero sé que están pensando en mí y deseándome lo mejor del mundo.

A mi novia Yaneisy por darme tantos consejos, apoyarme en los momentos buenos y malos, por enseñarme a querer, gracias por todo....

A mi tutora Alelí, gracias por todo, por haberme ayudado tanto, sin ti este momento no hubiese llegado, una vez más gracias

A mi familia por estar siempre presente cuando lo necesité.

A mis compañeros Ruben, Carlos y Novoa que desde primer año estuvieron ahí conmigo en cada trabajo o tarea, siempre de cara al estudio.

A mis compañeros Yasmani, Eider, Maisel, Barrueta, Adriel, Johan, Rodain, Lara, Nelly, Warrior, Tommy, etc. Por haber compartido conmigo ratos agradables.

A mi gente, mi novia Yaneisy, Yumi y Julio, Dunia, Rocko y Franco.

A mis amigos de toda la vida, aunque la universidad nos alejó un poco, pero nunca perdimos contacto, Maikel, Quintana, Ray y toda la gente del barrio.

Agradecerle a la revolución por haberme dado esta oportunidad, aunque sin la ayuda de mis padres no hubiese sido posible lograr este resultado.



Resumen

En la Universidad de las Ciencias Informáticas se prioriza la informatización de la gestión universitaria con el surgimiento del Centro de Informatización Universitaria. El mismo consta de proyectos que aseguran el funcionamiento del Sistema de Gestión Universitaria que abarca 12 áreas de procesos. En algunas de dichas áreas se realizan eventos de diferentes tipos. La planificación, coordinación, dirección y evaluación de los proyectos que en ellos se presentan es engorrosa debido a la no existencia de un sistema que sustituya el uso de herramientas ofimáticas y en el cual se encuentren informatizadas las funciones que implica la gestión de estos eventos. A pesar de que los procesos que incluyen algunos de los sucesos más importantes se encuentran informatizados, la falta de un lugar centralizado para el almacenamiento de su información y de una aplicación para la gestión de eventos independientemente de las reglas de negocio por las que cada uno se rija, son algunas de las causas por las que se ha decidido desarrollar una solución informática que permita gestionar información en los eventos como un proceso integrado, a partir de la configuración de las reglas del negocio para cada tipo de evento en el Sistema de Gestión Universitaria. La solución a la que se hace referencia maneja la información de eventos y proyectos que se presentan en ellos, las evaluaciones, y las estructuras que las emiten, así como la coordinación y planificación de dichos sucesos.

PALABRAS CLAVE

Evento, evaluación, gestión de eventos, gestión universitaria, proyectos.

Abstract

At the University of Informatics Sciences is prioritized informatization of university management with the advent of the University Informatization Center. It consists of projects which ensure the functioning of the University Management System covering 12 process areas. In some of these areas are made of different events. The planning, coordination, direction and evaluation of projects in these events is troublesome due to the absence of a system to replace the use of office tools and which are computerized functions involving the management of these events. Although the processes that include some of the most important events are computerized, the lack of a centralized location for storing information and an application for managing events independent of the business rules by which each is subject, are some of the reasons why it has decided to develop a software solution that allows to manage information in the events as an integrated process, from the configuration of business rules for each type of event into the University Management System. The solution referred manages information of events and projects presented in them, assessments, and the structures that produce it and coordination and planning of such events.

KEYWORDS

Assessment, event, event management, projects, university management.



Índice

Introducción	1
Capítulo 1: La gestión de eventos en la universidad y su informatización.	5
1.1 Introducción.....	5
1.2 La gestión de eventos en la universidad y el registro de información en los mismos	5
1.2.1 Evento en la gestión universitaria	5
1.2.2 Los proyectos en eventos en la gestión universitaria	5
1.2.3 Estructuras coordinadoras y evaluadoras de eventos en la gestión universitaria	7
1.3 Soluciones que informatizan la gestión de eventos en la gestión universitaria.	7
1.3.1 Soluciones extranjeras.....	7
1.3.2 Soluciones nacionales	12
1.3.3 Valoración crítica de los sistemas estudiados	15
1.4 Tendencias y tecnologías actuales.....	15
1.4.1 Modelos y técnicas de ingeniería de requisitos.	15
1.4.2 Patrones para el modelado de bases de datos.	17
1.5 Tecnologías del entorno de desarrollo.....	17
1.5.1 Proceso de desarrollo de software con enfoque ágil y CMMI nivel 2.	17
1.5.2 UML 2.0.....	18
1.5.3 BPMN 2.0	18
1.5.4 Visual Paradigm Suite 3.4.....	19
1.5.5 Netbeans IDE 7.1.....	19
1.5.6 PostgreSQL 8.4	20
1.5.7 GUUD.....	20
1.5.8 Apache 2.0.....	22
1.5.9 PgAdmin III 1.10.0	22
1.5.10 Lenguajes de programación.....	23
1.6 Conclusiones.....	24
Capítulo 2. Descripción de una propuesta de solución para la gestión de la información de eventos en la UCI.....	25
2.1 Introducción.....	25
2.2 Estudio preliminar.....	25
2.2.1 Modelado del negocio	26
2.3 Obtención y análisis de los requisitos de software.....	28
2.3.1 Objeto de informatización.....	28
2.3.2 Requisitos obtenidos.....	29
2.3.3 Audiencia del sistema	33
2.4.1 La solución como un todo	34
2.4 Especificación y validación de los requisitos.....	34
2.4.1 Especificación de requisitos	34
2.4.2 Validación de requisitos	37
2.5 Descripción de la arquitectura de sistema	38
2.5.1 Vista de presentación.....	39
2.5.2 Vista de integración.....	39
2.5.3 Vista de datos	41
2.5.4 Vista de despliegue.....	41
2.6 Conclusiones.....	42
Capítulo 3. Implementación y validación de la solución propuesta.....	43
3.1 Introducción.....	43

3.2 Técnicas de programación	43
3.2.1 Programación modular.....	43
3.2.2 Programación orientada a objetos (POO)	43
3.3 Estilos de programación	44
3.4 Estándares de codificación.....	44
3.4.1 Identación, llaves de apertura y cierre y tamaño de líneas.....	44
3.4.2 Conversión de nomenclatura	44
3.4.3 Estructuras de control	45
3.4.4 Documentación	45
3.5 Integración con el Sistema de gestión universitaria.	45
3.6 Interfaces propuestas	46
3.7 Pruebas.....	48
3.7.1 Niveles y técnicas de prueba	48
3.7.2 Métodos y técnicas de prueba	49
3.7.3 Estrategia de pruebas.	50
3.8 Aplicación y resultados de las pruebas.....	50
3.8.1 Pruebas de caja blanca.....	50
3.8.2 Pruebas de integración	54
3.8.3 Pruebas de sistema	59
3.9 Conclusiones.....	60
Conclusiones generales.....	61
Recomendaciones	62
Referencias Bibliográficas.....	63
Bibliografía consultada.....	66
Glosario de términos.....	67
Anexos.....	69
Anexo 1. Modelado de Procesos del Negocio con BPMN. Subproceso: Abrir evento.....	69
Anexo 2. Modelado de Procesos del Negocio con BPMN. Subproceso: Presentar y evaluar proyecto.	70
Anexo 3. Conceptualización inicial.	71
Anexo 4. Diagrama que indica cómo se gestiona un evento en el sistema.	71
Anexo 5. Diagrama que indica cómo se gestiona un tribunal y un comité en el sistema.	72
Anexo 6. Diagrama que indica cómo se registra un proyecto en el sistema.	73
Anexo 7. Diagrama que indica cómo se registra una evaluación para un proyecto en el sistema.	74
Anexo 8. Diagrama flujo de datos en la funcionalidad de gestionar un evento.	74
Anexo 9. Diagrama flujo de datos en la funcionalidad de gestionar tribunales y comités.	75
Anexo 10. Lista de verificación.	76
Anexo 10.1 Lista de verificación segunda iteración.....	78
Anexo 11. Mapa de navegación de la solución.....	79
Anexo 12. Modelo físico de la base de datos	81
Anexo 13. Código modificar comité	82
Anexo 14. Código modificar tribunal.....	82
Anexo 15. Diseños de casos de prueba de integración.	83
Anexo 15.1. Diseños de casos de prueba de integración caso prueba 2	84
Anexo 15.2. Diseños de casos de prueba de integración caso prueba 3.	86
Anexo 15.3. Diseños de casos de prueba de integración caso prueba 4.	88

Introducción

El continuo desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC) constituye un factor fundamental en el progreso de las distintas esferas de la sociedad. Las mismas han determinado la evolución y supervivencia de las personas, pues proporcionan disímiles beneficios en la salud, la educación y la economía a nivel mundial. Su auge ha estado acompañado por un gran avance en la gestión de la información y el conocimiento. En Cuba, con el desarrollo de las TIC, se han estado llevando a cabo nuevos proyectos en diferentes esferas de la sociedad. La universalización de la educación es un ejemplo actual de estas transformaciones, las cuales propiciaron la apertura de la Universidad de las Ciencias Informáticas (UCI). Esta se considera una universidad productiva, término que es utilizado en Cuba para caracterizar a una institución educacional capaz de relacionar el conocimiento universitario con la producción para la sociedad.

En la UCI existe la necesidad de informatizar los procesos sustantivos de la gestión universitaria. Esto tiene como fin mejorar el trabajo y con ello obtener resultados importantes en las diferentes áreas del quehacer universitario, así como satisfacer las necesidades informáticas que cada vez son mayores. La UCI cuenta con varios centros de producción dentro de los cuales se encuentra el Centro de Informatización Universitaria (CENIA). En el mismo existen proyectos que tributan a la creación de un Sistema de Gestión Universitaria (SGU), que permitirá organizar y controlar el trabajo en las 12 áreas de procesos de la Universidad: Ingreso, Pregrado, Investigaciones, Producción, Egreso, Laboratorios, Extensión universitaria, Teleformación, Biblioteca, Postgrado y Cooperación internacional. En la Extensión universitaria, en la Investigación y en el proceso docente de Pregrado y Postgrado se realizan eventos o actividades de diferente índole. Debido a la constante superación del personal en la UCI, la gestión de información durante el proceso de gestión de estos eventos se dificulta si no se apoya en sistemas o herramientas informáticas. Teniendo en cuenta esto se identificaron los siguientes elementos que conllevan a una situación problemática:

- La gestión de eventos en la Universidad de las Ciencias Informáticas tiene características comunes en cada uno de sus procesos sustantivos, sin embargo no existe una homogeneidad en la forma en que se desarrollan. Esto viene dado porque aunque todos los eventos que se realizan en la universidad tienen un mismo patrón o proceso general se aplican diferentes reglas de negocio para cada uno en particular.
- Para el desarrollo del proceso se utilizan herramientas ofimáticas que ralentizan el proceso no existiendo además un sistema único o genérico que permita realizar la gestión de los eventos que tienen lugar en la gestión universitaria, independientemente del tipo de evento que sea.

De lo anterior se puede deducir que la información no está centralizada provocando réplica de la misma, dificultando la caracterización de los eventos y el manejo de estadísticas confiables.

Es por esto que se identificó el siguiente **problema científico**: ¿Cómo realizar la gestión de la información de los eventos en la Universidad de las Ciencias Informáticas, como un proceso integrado, sin obviar las reglas del negocio para cada tipo de evento?

El **objeto de estudio** de la investigación que se presenta es el proceso de gestión de eventos en la gestión universitaria, definiendo como **campo de acción** la gestión de la información de eventos en la Universidad de las Ciencias Informáticas.

El **objetivo general** de este trabajo de diploma es desarrollar una solución informática para gestionar la información de los eventos universitarios, como un proceso integrado, a partir de la configuración de las reglas del negocio para cada tipo de evento en el Sistema de Gestión Universitaria.

Para garantizar el cumplimiento del objetivo general, el mismo se ha desglosado en los siguientes **objetivos específicos**:

- Caracterizar la gestión de eventos en la universidad y su informatización como fundamentación teórica de la investigación.
- Describir una propuesta de solución que pueda integrarse al Sistema de Gestión Universitaria permitiendo la gestión de la información de eventos en las áreas de procesos de la UCI.
- Implementar la solución propuesta utilizando las tecnologías y técnicas definidas.
- Validar la solución aplicando una estrategia de pruebas.

Para guiar la investigación se plantea como **idea a defender**: la utilización de una solución informática que permita la gestión de la información de los eventos como parte del Sistema de Gestión Universitaria, permitirá desarrollar dicho trabajo, en la Universidad de las Ciencias Informáticas, como un proceso integrado sin obviar las reglas del negocio para cada tipo de evento.

El **posible resultado** de esta investigación es la integración de la gestión de la información de los eventos, independientemente de su tipo, en una solución informática dentro del SGU; así como la centralización de la información registrada para los distintos eventos que se realizan en la Universidad de las Ciencias Informáticas facilitando la obtención de reportes y estadísticas confiables.

Para lograr dicho resultado se definieron las siguientes **tareas de investigación**:

- Caracterización de los procesos de gestión de eventos en las distintas áreas de procesos de la gestión universitaria a nivel nacional e internacional.
- Análisis de soluciones que informaticen los procesos de gestión de eventos en la gestión universitaria.



- Caracterización del proceso de desarrollo que se seguirá.
- Caracterización de las tecnologías que se emplearán en el entorno de desarrollo.
- Análisis de técnicas de ingeniería de requisitos.
- Obtención de los artefactos generados según el proceso de desarrollo aplicado.
- Definición de la estrategia de pruebas a emplear en la solución.
- Realizar el modelado de la base de datos y la descripción de la arquitectura.
- Implementación de los requisitos definidos.
- Validar la solución aplicando la estrategia de pruebas definida.

A continuación se muestran los métodos científicos utilizados para el desarrollo de esta investigación.

Métodos teóricos

Estos métodos permiten estudiar las características del problema que no son observables directamente.

Método histórico: se ha realizado un estudio de la trayectoria real del problema que se presenta tanto en el mundo como en Cuba.

Método lógico: basándose en el estudio del problema se expresa la esencia del mismo explicando su historia.

Método de la modelación: para la mejor comprensión del problema se han realizado abstracciones, utilizando modelos que tienen una estrecha relación con el que se estudia. Estos modelos representan la posible solución que tendrá el problema.

Análisis y síntesis: para un mejor entendimiento y estudio del problema se ha descompuesto el mismo en varios elementos, de esta forma se ha determinado cuáles son los fenómenos que lo generan. Luego de haber realizado el análisis de todos los fragmentos del problema se hace una unión del mismo posibilitando definir sus características generales y las relaciones que existen entre las partes analizadas.

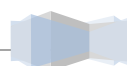
Métodos empíricos

Método de observación: con la utilización de este método es posible percibir directamente los hechos de la realidad objetiva y observar de forma abierta la población estudiada.

Revisión documental: se pone de manifiesto al revisar la bibliografía existente y consultar la información en internet para llevar a cabo las tareas de investigación.

Estructura del contenido

El trabajo consta de una introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía consultada y glosario de términos.



Capítulo 1. La gestión de eventos en la universidad y su informatización: se definen algunos conceptos importantes para la posterior comprensión de la investigación, se realiza una breve reseña y valoración sobre las soluciones existentes en Cuba y el mundo que informatizan la gestión de eventos en la gestión universitaria. Se tratan aspectos tales como las tendencias tecnológicas actuales, se explican las herramientas, tecnologías, técnicas, enfoques y modelos de desarrollo que serán empleados en el desarrollo de la solución que da nombre a esta investigación.

Capítulo 2. Descripción de una propuesta de solución para la gestión de la información de eventos en la UCI: en este capítulo se estudia el proceso de gestión de eventos en la gestión universitaria identificando aquellas actividades en las que se realiza gestión de información. A partir de esto se extrae el objeto de informatización. Al identificar las técnicas de ingeniería de requisitos que se emplearán se definen los requisitos funcionales y no funcionales. Luego se explican las vistas de presentación, integración, datos y despliegue que conformarán la arquitectura de la solución propuesta.

Capítulo 3. Implementación y validación de la solución propuesta: se describe la implementación del sistema propuesto teniendo en cuenta las técnicas de programación, estrategias de codificación, estándares y estilos, así como el tratamiento de errores y los patrones de diseño a utilizar. Se explica la integración de la solución con otros módulos y subsistemas del Sistema de Gestión Universitaria. Queda definida la estrategia de pruebas a emplear pasando a su posterior aplicación y manifestación de sus resultados.



Capítulo 1: La gestión de eventos en la universidad y su informatización.

1.1 Introducción

Luego del estudio de los diferentes procesos sustantivos de una universidad se identificarán aquellos donde está presente la gestión de eventos. A partir del desarrollo de las tecnologías este proceso ha sido informatizado en el mundo y en Cuba aunque en algunas instituciones no está definido como un proceso integrado dentro de la gestión universitaria. En este capítulo se pretende caracterizar la gestión de eventos y su informatización como fundamentación teórica de la investigación mediante algunos conceptos relacionados con el objeto de estudio, valoración de sistemas homólogos, tendencias, herramientas, tecnologías y técnicas.

1.2 La gestión de eventos en la universidad y el registro de información en los mismos

1.2.1 Evento en la gestión universitaria

El término evento proviene del latín *eventus* y de acuerdo al diccionario de la Real Academia Española (RAE), un evento es un suceso importante y programado, de índole social, académica, artística o deportiva (1). Celebrar un acontecimiento que implica planear, organizar, controlar y por último dirigir. Cualquier evento que se realiza en algún área de la gestión universitaria es considerado a los efectos de este trabajo como un evento en la gestión universitaria. En algunos procesos sustantivos dentro de la gestión universitaria en los cuales la gestión de eventos juega un papel fundamental. En la Formación académica de pregrado se lleva a cabo mediante la defensa de trabajos de diploma y organización de eventos docente-investigativos orientados a los estudiantes. En la Formación académica de postgrado se realizan tesis de maestría y doctorado. Por su parte en el área de Investigación se desarrollan eventos científicos de diversa índole en ocasiones con varios niveles de organización. La Extensión universitaria es otra área donde se gestionan eventos deportivos y culturales, los cuales tienen particularidades que los hacen diferentes del resto de los eventos de la universidad.

1.2.2 Los proyectos en eventos en la gestión universitaria

Según el diccionario de la Real Academia Española un proyecto es el primer esquema o plan de cualquier trabajo que se hace a veces como prueba antes de darle la forma definitiva (1).

Etapas de un proyecto:

La idea de proyecto: que consiste en establecer la necesidad u oportunidad a partir de la cual es posible iniciar el diseño del proyecto. La idea de proyecto puede iniciarse debido a alguna de las siguientes razones:

- Porque existen necesidades insatisfechas actuales o se prevé que existirán en el futuro si no se toman medidas al respecto.

- Porque existen potencialidades o recursos subaprovechados que pueden optimizarse y mejorar las condiciones actuales.
- Porque es necesario complementar o reforzar otras actividades o proyectos que se producen en el mismo lugar y con los mismos involucrados.
- Porque es necesario potenciar el interés en un campo determinado de la cultura y el deporte.

Diseño: etapa de un proyecto en la que se valoran las opciones, tácticas y estrategias a seguir, teniendo como indicador principal el objetivo a lograr. En esta etapa se produce la aprobación del proyecto, que se suele hacer luego de la revisión de la propuesta. Una vez dada la aprobación, se realiza la planificación operativa, un proceso relevante que consiste en prever los diferentes recursos y los plazos de tiempo necesarios para alcanzar los fines del proyecto, asimismo establece la asignación o requerimiento de personal respectivo.

Ejecución: consiste en poner en práctica la planificación llevada a cabo previamente.

Evaluación: etapa final de un proyecto en la que éste es revisado y se llevan a cabo las valoraciones pertinentes sobre lo planeado y lo ejecutado, así como sus resultados, en consideración al logro de los objetivos planteados.

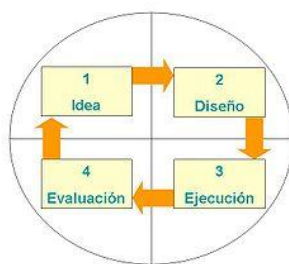


Figura 1 Etapas de un proyecto

A los efectos de esta investigación un proyecto puede ser:

Formación académica de pregrado: perfiles de trabajos de diploma y planillas de inscripción en eventos docente – investigativos.

Formación académica de postgrado: propuesta de trabajo final o tema de investigación.

Investigación: formulario de inscripción de trabajos a eventos.

Extensión universitaria: planilla de inscripción de un equipo en un deporte del evento, Planilla de inscripción de un número cultural al evento.



1.2.3 Estructuras coordinadoras y evaluadoras de eventos en la gestión universitaria

En la gestión universitaria se encuentran vigentes diversas estructuras cuya función es la de coordinar y evaluar los diferentes eventos que se realizan.

Los encargados de la organización o coordinación son los comités. En un primer nivel los eventos son coordinados por comités organizadores generales y en cada estructura inferior por comités organizadores de base.

A través de los comités se crean estructuras evaluadoras como los tribunales, los cuales pueden ser de varios tipos: tribunales de tesis, árbitros de juegos deportivos, jurado de festivales de artistas aficionados, tribunales de comisiones de eventos científicos, entre otros. Su función se centra en evaluar el resultado de las presentaciones de proyectos siempre que sea necesario. Dichas estructuras poseen características específicas para cada tipo de evento ya que deben cumplirse distintas reglas en la selección de sus miembros.

1.3 Soluciones que informatizan la gestión de eventos en la gestión universitaria.

La gestión de eventos está presente en varias áreas de la gestión universitaria. Muchas veces indispensables hasta en los renglones sociales, el auge de las tecnologías ha propiciado el surgimiento de disímiles sistemas que abarcan este proceso o parte de él. En este epígrafe se analizarán soluciones desarrolladas en el ámbito nacional e internacional con este fin y haciendo una breve valoración de las mismas.

1.3.1 Soluciones extranjeras

Sistema de Gestión de la Investigación (SGI):

Se ha desarrollado en la Universidad de Talca, Chile para apoyar las actividades académicas que realizan sus investigadores.

Los alcances o límites del SGI se enmarcan en el campo de la generación de conocimientos que se interrelacionan con subsistemas asociados al campo de la gestión académica y financiera. Según los autores de dicho sistema se identificaron estados para los proyectos de investigación y los procesos administrativos que los regulan. Las actividades físicas que producen los cambios de estado se definieron como sigue (2):

Postulación: actividad por la cual los proyectos que se formulan son registrados como postulantes a los distintos fondos concursables disponibles.” Evaluación: conjunto de actividades por las cuales los proyectos son revisados técnica y financieramente por quienes corresponda para su aprobación o rechazo.

Oficialización: actividad que se realiza sobre los proyectos aprobados para ser oficializados a través de convenios que posibilitarán la asignación de los recursos que su ejecución demanda.



Ejecución: conjunto de actividades asociadas al desarrollo del proyecto en base a lo comprometido al postular y que fuera aprobado en la instancia de la evaluación.

Cierre: actividad correspondiente a la rendición de cuentas y la emisión del informe final de los proyectos que contienen los resultados alcanzados, entre otros las publicaciones.

Las actividades administrativas asociadas a las actividades físicas identificadas son: (2)

Gestión de investigadores: conjunto de actividades responsables de informar a los investigadores de la oferta de fondos y sus características, además de monitorear su demanda por parte de los proyectos postulados por los investigadores.

Gestión de postulación: conjunto de actividades responsables de verificar el cumplimiento de las normas y requisitos formales exigidos a los proyectos, según los fondos a los que postulan y lo establecido por la Dirección de Programas de Investigación (DPI).

Gestión de evaluación: conjunto de actividades responsables de registrar y monitorear la evaluación de los proyectos postulados.

Gestión de oficialización: conjunto de actividades asociadas a la apertura de un proyecto aprobado que habilita la disponibilidad de los recursos aprobados.

Gestión de ejecución: conjunto de actividades responsables de respaldar el cumplimiento de los compromisos convenidos por los investigadores, en particular de la emisión de los correspondientes informes de avance.

Gestión de cierre: conjunto de actividades asociadas a la entrega de los informes finales por parte de los investigadores responsables de los proyectos.

En una primera etapa iniciada en el año 2005, se operó sobre una plataforma tecnológica basada en Oracle Workflow versión 2.6 con interface a la Intranet, desde donde se integra al SGI mediante un enlace a una opción que permite postular proyectos nuevos a los fondos disponibles. Esta modalidad de trabajo con Workflow fue presentada a la comunidad académica a fines del año 2005 (2)

Sistema de Administración Académica de Postgrado (SAACP)¹ desarrollado por el Centro de Servicios Informáticos (CSI) de la Escuela Superior Politécnica del Litoral de Ecuador (ESPOL).

El Sistema de Administración Académica de Postgrado, es un sistema de información desarrollado en Microsoft.Net, que permite apoyar el proceso de gestión académica de los programas a nivel de postgrados que se imparten en la ESPOL. La aplicación Windows está dirigida a las unidades académicas que dictan los programas de postgrados y administran la siguiente información:

¹ <http://www.csi.espol.edu.ec/ui/es/content/sistema/sistema.aspx?op=toshow&id=106>



- Proceso de graduación.
- Requisitos de grado.
- Pasantías.
- Tesis o proyecto de grado.
- Culminación de la malla curricular.
- Reportes varios.

Hércules²:

Sistema para la gestión de eventos deportivos desarrollado por la compañía Deporte Virtual pertenece al Información de Negocios y Procesos INP S.A., en Bogotá, Colombia capaz de:

- Registrar masivamente a los usuarios de los programas recreo-deportivos.
- Generar automáticamente reportes para la toma de decisiones gerenciales posibilitando conocer datos de usuarios beneficiados, participación, permanencia y rotación, además de los informes generales de usuarios por programa.
- Crear y controlar las etapas de un evento deportivo.
- Gestionar centros de entrenamiento incluyendo área médica, creación, seguimiento y consulta de entrenamiento.
- Recibir y coordinar las inscripciones de miles de usuarios conectados al tiempo con diferentes ubicaciones geográficas.
- La aplicación además cuenta con los módulos Eventos, Escuelas, Gimnasios, Entretenimiento que incluyen opciones como:
- Manejo de inscripciones: posibilita el control de manera sencilla de variables como integrantes máximos y mínimos de deportistas por equipos participantes, número de delegados, entrenadores, fisioterapeutas, fechas de inscripción, fechas de cambios y en general cualquier regla que se pueda presentar en un evento deportivo. Todo esto de forma independiente para cada deportista permitiendo que estos documentos se consoliden en una base de datos consultable por el comité organizador del evento en cualquier momento y desde cualquier computador con conexión a internet. El sistema permite fácilmente definir parámetros para controlar el proceso de inscripción como la configuración del formulario de inscripción en donde el organizador define los campos a solicitar y con sus reglas de inscripción, para cada evento.

² <http://hercules.deportevirtual.com/>



- Registro de escenarios: permite al organizador del evento registrar un listado de escenarios con datos generales como nombre, dirección, teléfono y contacto. Cada escenario en Hércules, podrá contar con reservas de tiempo, especificando rangos de fechas, días y horas en las que está disponible para la realización de los eventos deportivos.
- Voluntariado: brinda la opción de construir un formulario de inscripción personalizado de acuerdo sus requerimientos particulares. Se podrán incluir desde los datos generales, hasta particularidades como habilidades especiales de las personas y puestos de trabajo preferidos para trabajar en el evento. Todos los datos son parametrizables y condicionales según las políticas que tenga el comité organizador incluyendo fechas límites, puestos de trabajo, edades, características físicas, archivos adjuntos y en general cualquier particularidad.
- Reportes y estadísticas: brinda la opción de obtener en segundos, reportes de inscripción filtrados por deporte, categorías, roles, regionales y organizaciones participantes. Además muestra un resumen con los datos más usados en la organización de eventos deportivos.
- Planillas técnicas: las planillas técnicas son el documento que permite visualizar el consolidado de deportistas y personal de apoyo inscrito por delegación. Teniendo como base las inscripciones, Hércules puede generar automáticamente las planillas técnicas por delegación, deporte y género. Las planillas técnicas se pueden visualizar de diferentes maneras, de acuerdo con los permisos que tenga cada usuario en el sistema. Además posibilita exportar la planilla en formato xls.
- Acreditaciones: permite la generación automatizada de las acreditaciones de todo el evento. Estos documentos identificarán al participante durante el certamen, para contribuir a la organización y control de acceso a escenarios o recursos de los eventos.
- Resultados: posee una robusta herramienta que brinda la opción de registrar los resultados en tiempo real, directamente desde el escenario. Permite definir indicadores de competencia que procesan la información del deporte determinando con precisión el desarrollo de cada campeonato, ejemplo: Goles, Puntos, Sets, Tarjetas amarillas y rojas, Tiempos.
- Tabla de medallas: En los eventos manejados con Hércules la tabla de medallas se actualiza automáticamente, a medida que los resultados son ingresados por cada una de las personas encargadas en los escenarios.

E_eventos³

³ <http://indico.saber.ula.ve/eventos/>



Desde el año 2008 en la Universidad de Los Andes (ULA), Venezuela, se estableció el servicio e_eventos en donde la comunidad académica puede publicar los distintos eventos, congresos, talleres, coloquios, simposios, encuentros, etc, a través de herramientas de gestión de eventos desarrolladas en software libre. Esta iniciativa permite la promoción y la visibilidad en la WEB de eventos académicos. La aplicación permite planificar un evento de acuerdo a un calendario publicando el mismo en una categoría. Los organizadores del evento pueden gestionar el contenido a mostrar en la página del mismo como por ejemplo: Comité organizador, auspiciantes, inscripción, temáticas, pero solo de manera informativa no permitiendo en registro de información por parte del usuario.

Open Conference Systems (OCS) (3)

Open Conference Systems (OCS) es una aplicación gratuita y de código abierto para la gestión y publicación de congresos académicos en la web. Al tratarse de un sistema de gran flexibilidad, es posible generalizar los congresos a eventos académicos de todo tipo, así como a estudios académicos en general en todos los niveles, tanto en Educación Primaria y Secundaria, Bachiller, como en Educación Superior (Grado, Máster y Doctorado) o cualquier otro tipo de evento académico (Cursos, Jornadas, Seminarios, Workshops, etc.), por lo que es posible tratar con trabajos académicos en general, generalizando desde las actas y presentaciones propias de un congreso, aprovechando todas las funciones y características que ofrece este sistema .

Esta aplicación permite a la institución llevar un registro, un archivo y un histórico de todos los trabajos académicos desarrollados por los alumnos, realizando una gestión de los mismos totalmente transparente. El profesorado puede crear y desarrollar con sus alumnos un espacio colaborativo y participativo en la red con el que trabajar en la gestión, evaluación y publicación de los trabajos académicos a través de la web. Para obtener una cuenta de usuario en el sistema sólo es necesario disponer de una cuenta de correo electrónico.

Con el uso de OCS en el aula, se pueden fijar otro tipo de objetivos o desarrollo de competencias como, por ejemplo, iniciar a los alumnos en la investigación, adoptando el modelo de presentación y evaluación de trabajos de las revistas científicas y congresos, aprovechando la evaluación por pares que ofrece el sistema. Esto cobra más importancia en la Educación Superior y además da lugar a emprender iniciativas de iniciación a la actividad investigadora para los alumnos de secundaria.

Otra de las características que puede llegar a ser muy ventajosa, una vez implementada la aplicación en el proceso educativo, es que se puede organizar la presentación de trabajos finales de asignaturas, trabajos y proyectos de fin de curso, proyectos finales de carrera, tesis de máster, tesis doctorales, etc., a través de un modelo de evento académico como puede ser un simposio o un congreso.



1.3.2 Soluciones nacionales

En materia de informática Cuba no tiene un amplio desarrollo debido a que aún quedan muchos procesos de la sociedad cubana por informatizar y a la falta de recursos materiales para desarrollarse tecnológicamente, pero a medida que pasa el tiempo se avanza por obtener una soberanía tecnológica e informatizar la sociedad cubana.

UCIENCIA⁴

La Conferencia Científica UCIENCIA es un evento que se realiza en la UCI desde el 2005. Para la IV edición se utilizó una aplicación web como modalidad virtual de la misma. Dicha aplicación contiene entre sus principales funcionalidades la creación del programa del evento, así como el registro de talleres, conferencias y ponencias a presentar. Además posibilita registrar la información referente a los comités científico y organizador.

Es un sitio interactivo que facilita la consulta de las ponencias presentadas a partir de un buscador central y la emisión de comentarios sobre las mismas. Los usuarios pueden además consultar reportes estáticos sobre el evento. De la misma forma pueden consultar su perfil para ver el historial de acciones en el sistema y sus envíos de ponencias al evento así como las distintas revisiones de los trabajos inscritos por el usuario.

Aplicación Web para el control del Proceso de Tesis de Grado en la Facultad 8, UCI.

Esta es una aplicación web que maneja información de los documentos generados por el proceso de tesis de grado y las autoridades concernientes en la antigua Facultad 8, hoy Facultad 4, de la UCI. La misma fue el resultado de la investigación en un trabajo de diploma en el año 2008.

Fue desarrollada con el CMS Drupal e informatiza el trabajo de las entidades relacionadas con el proceso de desarrollo de tesis de grado así como la centralización de los documentos generados referentes al proceso en cuestión. Esta aplicación posibilita publicar los temas de tesis propuestos por los profesores, asignar dichos temas a los estudiantes de quinto año que aun no tienen un tema asociado, tener un seguimiento de la evaluación y evolución de las tesis así como de sus autores. Facilita la obtención de reportes relacionados con todo el proceso de tesis, que pueden ser, entre otros, horarios de exposición, locales, tesis evaluadas, tesis sin evaluar, tribunales, tutores, etc. (4).

Producto Congreso en Informática en Salud 2009 (5)

Este producto se hizo a partir de una modificación de la herramienta Plone. Esta es un sistema de gestión de contenidos potente, que funciona sobre la plataforma del servidor de aplicaciones web Zope

⁴ <http://uciencia.uci.cu/>



y utiliza el modelo de metadatos Dublin Core. Permite que la información se coloque en los primeros lugares en la recuperación que realizan los buscadores en Internet como Google.

Plone está construido sobre una arquitectura de componentes reutilizables. Se puede trabajar con independencia del sistema operativo (Windows, Linux, Mac OS X, BSD, Solaris). Todos los elementos del sistema de información presentan flujo de trabajo (workflow), incluso los usuarios.

El usuario del sitio puede inscribirse y, una vez que verifica su autenticación, acceder al conjunto de servicios. La interfaz de Plone es compatible con los estándares de las cascadas de estilo (CSS) y códigos de HTML y esto permite modificarla con facilidad. La administración de esta herramienta es sencilla y posee diferentes niveles de acceso a la información, en dependencia de las funciones de los usuarios. Estas funciones se asignan en la administración de usuarios y grupos en Zope:

Administrador de Zope: tiene todos los permisos sobre los sitios en la instancia.

Administrador o manager: es aquella persona que designa el comité organizador para administrar el sitio, supervisar la actualización del contenido y el buen funcionamiento del servicio. Tiene los permisos de modificación del contenido del sitio que administra.

Revisor: revisa los contenidos añadidos por los miembros en el sitio y concede el permiso de publicación. Solo ve el contenido si el miembro envía el trabajo.

Miembro: es el participante activo del evento, quien envía un trabajo para publicar o participar en las modalidades disponibles en el sitio. Se inscribe en el sitio y se autentica. Solo puede añadir contenido en su carpeta o área de trabajo y puede interactuar con otros miembros por medio de comentarios.

Revisor: un miembro (o más) del evento que recibe el contenido enviado por los miembros o delegados (en esta ocasión los trabajos o ponencias que participarán en el evento). Una vez que comprueba que la información cumple los requerimientos para el evento, envía estos trabajos a los miembros del comité científico para su revisión.

Anónimo: tiene acceso a toda la información publicada en el sitio, pero no puede colocar trabajos en él. En el VI Congreso Internacional de Informática en Salud, realizado en 2007, fue necesario añadir al flujo de trabajo de Plone, nuevas funciones y facilidades a algunas de las existentes:

Miembro: se le restringió el *Trabajo* como único tipo de contenido que puede subir. Para que su trabajo pueda ser revisado, debe enviarlo desde la opción *Estado* disponible en el menú de edición del trabajo. En esta opción solo puede enviar el trabajo o ponerlo privado. Una vez enviado el trabajo, queda pendiente de revisión en el sitio y el autor no lo puede editar o retirar hasta que el revisor o comité científico le permita modificarlo. En el menú de edición de *Mi carpeta*, el miembro no debe tener acceso a *Propiedades*, *Traducir a*, *Acciones* y *Avanzado*.

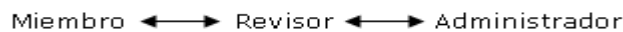


Comité científico: grupo de administración añadido al Plone, para un grupo de especialistas que revisa, evalúa, aprueba o rechaza el contenido científico de los trabajos que participarán en el evento. En caso de ser aprobado, otorga el permiso para que se publique el trabajo en el sitio y puede proponer la participación presencial de algún delegado, en caso que sea pertinente. Propone los miembros o especialistas que impartirán conferencias en el evento (conferencista).

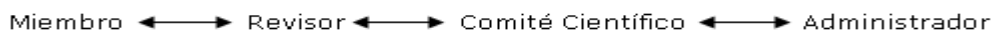
Conferencista: nueva función en el sitio para el especialista que imparte conferencias. Tiene la posibilidad de subir la presentación en los tipos de contenido: artículo, página o artículo multipáginas, según convenga. En la práctica, una vez impartida la conferencia, el administrador del sitio la coloca en la carpeta de *Conferencias*, en el formato entregado por el conferencista.

Fotógrafo: otra nueva función que solo tiene permiso para subir imágenes relacionadas con el evento, por ejemplo, la inauguración, la clausura y las sesiones de trabajo.

El flujo de la información en un sitio web original en Plone es:



En este evento se añadió un nivel intermedio entre el revisor y el administrador y a este se le llamó comité científico quedando de la manera siguiente:



Este flujo de información comienza cuando el miembro del sitio (delegado) coloca el trabajo en su carpeta, modifica el estado y lo envía al revisor. Este lo envía al comité científico, quien lo publica o suspende. Los contenidos pueden permanecer en estado privado o en borrador público y el miembro determinar en qué fecha o rango de fechas lo publica (figura 2), que es cuando la información se puede recuperar.

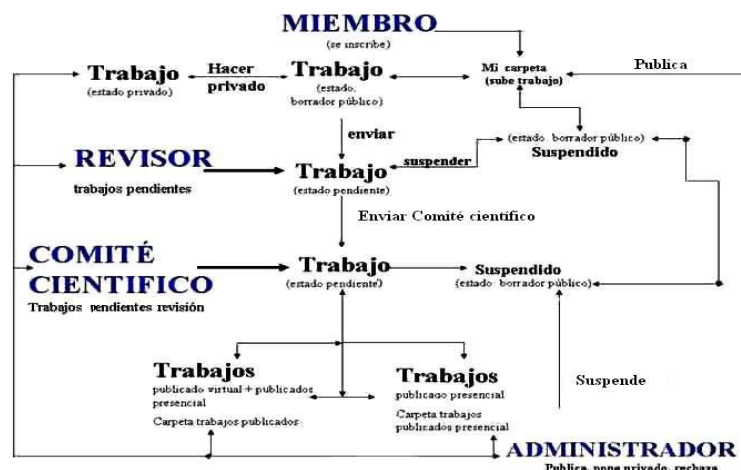


Figura 2 Flujo de información en los trabajos



1.3.3 Valoración crítica de los sistemas estudiados

Luego de un análisis de estos sistemas se dedujo que a pesar de que ninguno brinda una solución completa, se han identificado, en algunos de ellos, características reutilizables en el desarrollo de la solución que se pretende obtener. Elementos de los sistemas estudiados como la idea de gestión de los proyectos y actualización de sus estados, la evaluación de dichos proyectos y la gestión de ejecución como planificación de los eventos serán aplicadas en la propuesta a desarrollar. Al estudiar estos sistemas es posible percatarse de que los eventos se dividen en dos grandes ramas: Eventos investigativos y Eventos socioculturales. Los sistemas que informatizan los primeros garantizan la gestión y revisión además de los informes que exponen el resultado del proyecto así como las líneas y temas posibles a investigar. Por su parte los sistemas que informatizan los segundos se enfocan más en la planificación, organización y logística del evento que en los proyectos y resultados. Es por esto que se decide buscar los puntos en común de estos tipos de eventos para obtener una solución genérica que además, de forma particular, permita satisfacer las necesidades de gestión, registro de información y control de ambos tipos de eventos.

1.4 Tendencias y tecnologías actuales

1.4.1 Modelos y técnicas de ingeniería de requisitos.

Todo proceso de desarrollo de software necesita de una organización y una estructura, brindada por disímiles modelos que se ajustan en dependencia del software que se quiere construir. Los modelos de proceso enfatizan la definición, la identificación y la aplicación detallada de las actividades del proceso. En los últimos años se han propuesto modelos de procesos acompañados de un enfoque ágil en busca de la adaptabilidad, los mismos son apropiados para la construcción de aplicaciones en red. Junto a ellos se tiende a la aplicación de una ingeniería de requisitos con mayor consistencia.

La Ingeniería de Requisitos (IR) consiste en un conjunto de actividades y transformaciones que pretenden comprender las necesidades de un sistema de software y convertir la declaración de estas necesidades en una descripción completa, precisa y documentada de los requisitos del sistema siguiendo un determinado estándar (6). Actividades propias de esta área, como la captura de requisitos del usuario, la especificación de requisitos o la validación de los mismos, son algunas de las consideradas más críticas en el desarrollo y la producción del software.

Dentro de la captura de requisitos se pueden identificar técnicas como: (7)

- Entrevista: las entrevistas le permiten al analista tomar conocimiento del problema y comprender los objetivos de la solución buscada.
- Tormenta de ideas (Brainstorming): se generan ideas que son criticadas y analizadas libremente por el grupo. Ayuda a obtener diferentes enfoques del problema y plantear soluciones de diversas



formas. Permite obtener una visión general y clara de las necesidades del sistema o proyecto, pero no se puede obtener suficiente detalle sobre los mismos.

También para la actividad de especificación de requisitos en el proceso de ingeniería de requisitos hay un gran número de técnicas propuestas: (7)

- Glosario y ontologías: aboga por desarrollar un glosario de términos en el que se recogen y definen los conceptos más relevantes para el sistema. En esta línea se encuentra también el uso de ontologías, en las que no sólo aparecen los términos, sino también las relaciones entre ellos.
- Plantillas o patrones: esta técnica, tiene por objetivo el describir los requisitos mediante el lenguaje natural pero de una forma estructurada a través del uso de plantillas.
- Escenarios: la técnica de los escenarios consiste en describir las características del sistema a desarrollar mediante una secuencia de pasos.
- Prototipos: algunas propuestas se basan en obtener de la definición de requisitos prototipos que, sin tener la totalidad de la funcionalidad del sistema, permitan al usuario hacerse una idea de la estructura de la interfaz del mismo.
- Los requisitos una vez definidos necesitan ser validados. Para ello también se cuenta con diversas técnicas: (8)
- Revisiones de requerimientos: comprueban consistencia, integridad, verificabilidad, comprensibilidad y rastreabilidad. Proceso manual que involucra a varios lectores, tanto del cliente como del contratista. Puede ser formal o informal, el equipo de desarrollo conduce al cliente a través de los requerimientos, explicándole las implicaciones de cada uno.
- Construcción de prototipos: consiste en construir una maqueta del futuro sistema de software a partir de los requisitos recogidos en la especificación. Esta maqueta será evaluada por el cliente y usuarios para comprobar su corrección y completitud.
- Generación de casos de pruebas: tiene como objetivo comprobar la verificabilidad de los requisitos. Consiste en la definición de casos de prueba que permitan verificar el cumplimiento de los requisitos funcionales.

A partir del estudio de las técnicas antes mencionadas se han seleccionado la entrevista y la tormenta de ideas para el desarrollo de la actividad de captura. Además se ha definido que el uso de prototipos, escenarios y plantillas resultan las opciones más eficientes en la especificación de los requisitos que se definirán para la solución. Finalmente se ha identificado la técnica de revisión de requisitos y la generación de casos de prueba hacia la validación eficaz de la propuesta de solución.



1.4.2 Patrones para el modelado de bases de datos.

Un patrón de diseño de bases de datos es una plantilla que ya ha sido evaluada como la responsable de resolver un problema, es una guía para apoyarse en realizar el trabajo. Le permiten al usuario crear una base de datos más fortalecida ya que tiene una guía que especifica cómo debe ser la base de datos (9).

Modelo entidad-atributo-valor: el modelo entidad-atributo-valor es la representación de un modelo flexible donde se pueden representar objetos con sus atributos, es un acercamiento al modelo orientado a objeto representado en el modelo relacional, donde la entidad Class representa las clases, la entidad Attribute representa los atributos de las clases, por su parte la entidad Object representa las instancias de las clases, mientras que la entidad Value representa los valores de cada atributo para cada objeto dado (9).

1.5 Tecnologías del entorno de desarrollo

En este epígrafe se describen brevemente las tecnologías que serán utilizadas para el desarrollo de la solución a partir de los estudios realizados y los requerimientos definidos por el CENIA para el desarrollo del Sistema de Gestión Universitaria.

1.5.1 Proceso de desarrollo de software con enfoque ágil y CMMI nivel 2.

El proceso de desarrollo de software no es único. No existe un proceso de software universal que sea efectivo para todos los contextos de proyectos de desarrollo. Debido a esta diversidad, es difícil informatizar todo un proceso de desarrollo de software (9).

Dado que el software es inmaterial, la calidad de software es intangible, pero a pesar de esto se tienen ciertas pautas para determinar la calidad de un determinado software, entre estas se encuentra: (10)

- Que el software en cuestión, se acerque a tener cero defectos.
- Que se haya cumplido con todos los requisitos intrínsecos y expresos.
- Que se logre alcanzar la satisfacción del cliente.

La calidad de software es el conjunto de cualidades que caracterizan a un software y que determinan su utilidad y existencia. La calidad es sinónimo de eficiencia, flexibilidad, corrección, confiabilidad, mantenibilidad, portabilidad, usabilidad, seguridad e integridad (10).

Un modelo de calidad de software es un conjunto de buenas prácticas para el ciclo de vida del software, enfocadas en los procesos de gestión y desarrollo de proyectos, este indica qué hacer y no cómo hacerlo. Es el conjunto de características y las relaciones entre las mismas, que proveen la base para especificar requisitos de calidad y evaluar la calidad. El Modelo Integrado de Capacidad y Madurez (CMMI) para desarrollo contempla las buenas prácticas relativas a las actividades de desarrollo y mantenimiento aplicadas a productos y servicios. Trata las prácticas que cubren el ciclo de



vida del producto desde la concepción hasta la entrega y el mantenimiento. En el nivel de madurez 2, los proyectos de la organización han asegurado que los procesos se planifican y realizan de acuerdo a políticas; los proyectos emplean personal con habilidad que dispone de recursos adecuados para producir resultados controlados; involucran a las partes interesadas relevantes; se monitorizan, controlan y revisan; y se evalúan en cuanto a su adherencia a sus descripciones de proceso. (11).

En marzo de 2001 diecisiete críticos de los modelos de mejora del desarrollo de software basados en procesos, convocados por Kent Beck, quien había publicado un par de años antes Extreme Programming Explained, se reunieron para tratar sobre técnicas y procesos para desarrollar software. Los integrantes de la reunión resumieron los principios sobre los que se basan los métodos alternativos en cuatro postulados, lo que ha quedado denominado como Manifiesto Ágil (12).

- Valorar más a los individuos y su interacción que a los procesos y las herramientas.
- Valorar más el software que funciona que la documentación exhaustiva.
- Valorar más la colaboración con el cliente que la negociación contractual.
- Valorar más la respuesta al cambio que el seguimiento de un plan.

CMMI es un modelo de mejores prácticas traducidas en “metas” que indican “qué” hacer. Cada meta tiene una serie de “prácticas” a cubrir. Ágil es un enfoque de desarrollo que propone “prácticas” enfocadas en “cómo” hacer las cosas.

1.5.2 UML 2.0

Los lenguajes de modelado son un conjunto estandarizado de símbolos y sus relaciones que permiten expresar un sistema de información mediante un esquema teórico, el cual representará su diseño. El modelado visual permite manejar la complejidad de los sistemas a analizar o diseñar. Un modelo es expresado en un lenguaje de modelado compuesto por vistas, elementos de modelos y un conjunto de mecanismos generales o reglas que indican cómo utilizar los elementos (13).

UML es un lenguaje para especificar, construir, visualizar y documentar los artefactos (información que se utiliza o produce mediante un proceso de software). Este lenguaje de modelado no es una guía para realizar el análisis y diseño orientado a objeto, es decir, no es un proceso, es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos. Su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que UML ha sido diseñado para modelar cualquier tipo de proyecto informático, de arquitectura o cualquier otra rama (13).

1.5.3 BPMN 2.0

La Notación para el Modelado de Procesos de Negocio (BPMN), como lo indica su nombre, no es otra cosa que una notación gráfica estandarizada para el modelado de los procesos de negocio. Su



principal objetivo es resolver las dificultades de comunicación que tiene el lenguaje común (14). BPMN además:

- Proporciona un método normalizado para representar procesos de negocio
- Facilita su entendimiento debido a la poca complejidad de su notación
- Proporciona un lenguaje común entre los usuarios de negocio y los técnicos
- Facilita la diagramación de los procesos de negocio

Está basado en la idea de que existen diferentes roles con diferentes responsabilidades y cuya comunicación se realiza de forma similar a lo que sería un diagrama de estados pero dentro de un tiempo determinado (14).

1.5.4 Visual Paradigm Suite 3.4

Las herramientas CASE son un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software (15).

Visual Paradigm es una de las herramientas UML-CASE del mercado, considerada como muy completa y fácil de usar, con soporte multiplataforma y que proporciona excelentes facilidades de interoperabilidad con otras aplicaciones. Fue creada para el ciclo vital completo del desarrollo de software que lo automatiza y acelera, permitiendo la captura de requisitos, análisis, diseño e implementación. Tiene la capacidad de crear el esquema de clases a partir de una base de datos y crear la definición de base de datos a partir del esquema de las clases. Permite invertir código fuente de programas, archivos ejecutables y binarios en modelos UML al instante, creando de manera simple toda la documentación. Está diseñada para usuarios interesados en sistemas de software de gran escala con el uso del acercamiento orientado a objetos, además apoya los estándares más recientes de las notaciones de Java y de UML. Incorpora el soporte para trabajo en equipo, que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros (9).

1.5.5 Netbeans IDE 7.1

Netbeans es un entorno integrado de desarrollo libre y de código abierto para desarrolladores de software. Contiene todas las herramientas necesarias para crear aplicaciones profesionales de escritorio, empresariales, web y aplicaciones móviles con la plataforma Java, así como con C / C + +, PHP, JavaScript y Groovy (16).

Netbeans IDE 7.1 introduce soporte para JavaFX 2.0, así como herramientas para la depuración visual de Swing y las interfaces de usuario JavaFX. Otros puntos destacados incluyen el apoyo Git integrado en el IDE, las nuevas características de depuración de PHP, Java EE y diversas mejoras en Maven.



Netbeans es una plataforma que contiene un IDE de desarrollo, la cual permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases escritas para interactuar con las APIs de Netbeans y un archivo especial que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma Netbeans pueden ser extendidas fácilmente por otros desarrolladores de software (16).

El editor de Netbeans PHP ofrece plantillas de código y generación (getters y setters), como parte de la refactorización, información sobre herramientas de parámetros, consejos y soluciones rápidas (ej: implementar todos los métodos abstractos) y la finalización de código inteligente (incluida la finalización del soporte). Disfrute de resaltado de código sintáctico y semántico, la documentación emergente, formateo de código y plegado y el marcado de los sucesos y puntos de salida (16).

La depuración de código PHP es con Xdebug. Permite inspeccionar las variables locales, establecer inspecciones, puntos de interrupción y evaluar código en tiempo de ejecución. Netbeans proporciona soporte básico para FTP/SFTP, suficiente para que un desarrollador independiente pueda trabajar en un proyecto simple. El IDE también admite una amplia gama de sistemas de control de versiones incluyendo CVS, SVN, Git y Mercurial (16).

1.5.6 PostgreSql 8.4

Los Sistemas Gestores de Bases de Datos (SGBD) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, un lenguaje de manipulación de datos y un lenguaje de consulta. Entre los sistemas gestores de bases de datos más usados en la actualidad se encuentran MySQL, Oracle, SQL Server, PostgreSql, entre otros (17).

PostgreSql es un sistema gestor de bases de datos objeto-relacional que soporta gran parte del estándar SQL. Cuenta con características avanzadas como consultas complejas, llaves foráneas, disparadores, vistas, integridad transaccional, control de concurrencia multiversión. Puede ser extendido por el usuario añadiéndole tipos de datos, funciones, operadores, métodos de indexado y lenguajes procedurales. Funciona sobre varias plataformas incluyendo Windows XP, Linux, FreeBSD, Solaris y Unix. También soporta gran cantidad de lenguajes para el desarrollo de las aplicaciones como SQL, Java, Perl, Python, C, C++, Ruby y PHP. Además PostgreSql cuenta con una comunidad de desarrollo y soporte distribuida por todo el mundo (18).

1.5.7 GUUD

Gestión Universitaria de la Universidad Digital (GUUD) es un marco de trabajo propuesto por el equipo



de arquitectura del CENIA. El mismo integra a su vez los frameworks CodeIgniter 1.7.3 y JQuery 1.3.2 en una sola infraestructura, razón por la cual posee las mismas características que estos. En esta integración se incluyen además un conjunto de novedades o mejoras y algunas modificaciones hechas específicamente al CodeIgniter que se explican en el **acápito 1.5.7.1**

CodeIgniter es un marco de trabajo para el desarrollo de aplicaciones web escritas en PHP. Este permite el desarrollo de proyectos mucho más rápido que si se escribiera código desde cero. Provee una amplia colección de librerías para las tareas necesarias más comunes. Permite concentrarse en el desarrollo del proyecto en cuestión, minimizando la cantidad de código necesaria para realizar las tareas. CodeIgniter usa el patrón de diseño arquitectónico Modelo-Vista-Controlador como paradigma de arquitectura de desarrollo, la cual separa en 3 capas distintas la representación de datos, la interfaz de usuario y el controlador de eventos respectivamente (19).

El jQuery es un nuevo tipo de biblioteca o marco de trabajo de JavaScript que permite acceder a los objetos del DOM de un modo simplificado. JQuery al igual que otras librerías, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código. Es decir, con las funciones propias de esta librería se logran grandes resultados en menos tiempo y espacio. El gran aporte de JQuery es que permite cambiar el contenido de la página web sin necesidad de recargarla, utilizando DOM y AJAX de manera sencilla gracias a su sintaxis (20)

1.5.7.1 Novedades que incorpora el GUUD

A continuación se muestra una relación de las principales mejoras y modificaciones que incorpora el GUUD en su infraestructura.

Del lado del cliente:

1. Se implementaron una serie de widgets para utilizarlos de interfaz de algunos de los posee jquery -ui como por ejemplo el date, el tab (ambos son interfaces de los widgets de mismo nombre de jquery-ui) y el popup (interfaz del dialog de jquery-ui). Además de los ya mencionados se implementaron otros nuevos entre los que se encuentran: attach, menú, message, tooltip, form (se construyó con la unión de los plugins form de JQuery el cual se utiliza para el envío de formularios AJAX y el validate utilizado para validar formularios), grid (utiliza como plugin el jqgrid), multiselect (para hacer selecciones múltiples), navbar (para la creación de barras de navegación), tree (para la creación de árboles) y el graph (utiliza la librería Highchart).
2. Se implementó un plugin para JQuery 1.3.2 que permite el manejo de espacios de nombre e internacionalización.
3. Se implementaron funciones comunes para todo el sistema (contenidas en los archivos core.js y common.js) entre las que se destacan: loadIn, getDataJson, createSelect, isArray, isFunction, site_url



Del lado del servidor (incorporadas a CodeIgniter):

1. Se agregó el manejo de excepciones y mensajes.
2. Se implementó el IOC (inversion of control) para la interacción entre módulos.
3. Se añadió la característica de la modularidad o sea que una aplicación pueda dividirse en módulos.
4. Se añadieron, modificaron y extendieron los helpers o asistentes entre los que se encuentran:
 - Añadidos: `template` (brinda la posibilidad de usar plantillas, característica que no posee CodeIgniter. Para esto se añadió también la librería `template`), `assets` (utilizado para la integración en las vistas de javascript, css, imágenes y el `template`), `grid`, `json`.
 - Modificados: `form`, `array`, `security`.
5. Se añadieron los plugins `export_pi` (permite exportar a los formatos: pdf, csv y xls) e `import_pi` (permite importar desde archivos en formatos csv o xls).

1.5.8 Apache 2.0.

Apache 2.0: es un servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos (HTTP 1.1). Es un servidor web altamente configurable, robusto y estable. Es un sistema de código abierto para plataformas Windows, Unix, Macintosh y otras que implementa el protocolo HTTP. Entre sus características se destacan (21):

- Multiplataforma
- Es un servidor de web conforme al protocolo HTTP/1.1
- Modular ya que puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona y con la API de programación de módulos, para el desarrollo de módulos específicos.
- Basado en hebras en la versión 2.0
- Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos.
- Se desarrolla de forma abierta
- Extensible pues gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP.

1.5.9 PgAdmin III 1.10.0

PgAdmin III es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, siendo la más completa y popular con licencia Open Source. Es capaz de gestionar versiones a partir de la PostgreSQL 7.3 .1 está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. El interfaz gráfico soporta

todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis (22).

1.5.10 Lenguajes de programación

Java Script (23).

Java Script es un lenguaje de script multiplataforma [cross-platform] orientado a objetos. Es un lenguaje pequeño y ligero que no es útil como un lenguaje independiente, más bien está diseñado para una fácil incrustación en otros productos y aplicaciones, tales como los navegadores Web. Dentro de un entorno anfitrión, Java Script puede ser conectado a los objetos de su entorno para proveer un control programable sobre éstos. El núcleo de Java Script contiene un conjunto central de objetos, tales como Array (arreglos), Date (fechas) y Math (objetos matemáticos), además de un conjunto central de elementos del lenguaje tales como los operadores, estructuras de control y sentencias. El núcleo de Java Script puede ser extendido para una variedad de propósitos complementándolo con objetos adicionales, por ejemplo:

- *Java Script del lado Cliente* extiende el núcleo del lenguaje proporcionando objetos para el control del navegador y su Modelo Objeto Documento (DOM). Las extensiones del lado del cliente permiten a una aplicación ubicar elementos en un formulario HTML y responder a los eventos de usuario tales como los clics del mouse, entradas del formulario y navegación de páginas.
- *Java Script del lado Servidor* extiende el núcleo del lenguaje proporcionando objetos relevantes para la ejecución de Java Script en un servidor. Por ejemplo, las extensiones del lado del servidor permiten que una aplicación se comunique con una base de datos relacional, proporcione continuidad de la información desde una invocación de la aplicación a otra o efectúe la manipulación de archivos en un servidor.

CSS 2 (24).

Hojas de Estilo en Cascada (Cascading Style Sheets) es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos. CSS se utiliza para dar estilo a documentos HTML y XML, separando el contenido de la presentación. Los *Estilos* definen la forma de mostrar los elementos HTML y XML. CSS permite a los desarrolladores web controlar el estilo y el formato de múltiples páginas web al mismo tiempo. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento. CSS funciona a base de reglas, es decir,



declaraciones sobre el estilo de uno o más elementos. Las hojas de estilo están compuestas por una o más de esas reglas aplicadas a un documento HTML o XML. La regla tiene dos partes: un selector y la declaración. A su vez la declaración está compuesta por una propiedad y el valor que se le asigne.

PHP 5.3 (25).

Personal Home Page (PHP) es un lenguaje de código abierto muy popular especialmente adecuado para desarrollo web y que puede ser incrustado en HTML. PHP puede procesar la información de formularios, generar páginas con contenidos dinámicos, enviar y recibir cookies. También puede ser utilizado en cualquiera de los principales sistemas operativos del mercado basados en Unix y MS-DOS. PHP soporta la mayoría de los servidores web de hoy en día, incluyendo Apache. PHP funciona ya sea como un módulo o como un procesador de CGI. Una de las características más potentes y destacables de este lenguaje es su soporte para una gran cantidad de bases de datos. Escribir una página web con acceso habilitado a una base de datos es increíblemente simple utilizando una de las extensiones específicas, utilizar una capa de abstracción como PDO o conectarse a cualquier base de datos que soporte el estándar de Conexión Abierta a Bases de Datos por medio de la extensión ODBC. Otras bases de datos podrían utilizar URL o sockets como lo hace CouchDB. PHP también cuenta con soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM entre los principales.

1.6 Conclusiones

Como parte del análisis de la situación problemática planteada, se han definido conceptos fundamentales relacionados con la gestión de eventos y su informatización. Se realizó un estudio de aplicaciones homólogas tanto nacionales como extranjeras entre las que se destaca el Hércules, el Open Conference Systems, el sitio web de UCIENCIA y el producto Congreso realizado en el 2009, por poseer un valor agregado a la solución que se pretende obtener. Además se han descrito las herramientas, técnicas y lenguajes establecidos por las políticas del centro CENIA. De esta forma quedan caracterizados los principales elementos de la fundamentación teórica de esta investigación.



Capítulo 2. Descripción de una propuesta de solución para la gestión de la información de eventos en la UCI

2.1 Introducción

Conocer el funcionamiento del negocio y sus reglas es imprescindible en la construcción de un software efectivo. A partir de ello se concretan las necesidades del cliente. En este capítulo se describirán el proceso vigente de la gestión de eventos en la gestión universitaria. Al modelar este proceso se identificaron aquellas actividades donde se realiza gestión de información y las reglas que deben cumplirse dado un evento cualquiera. Esto impulsó la especificación de requerimientos funcionales y no funcionales y la descripción de una arquitectura que permita la futura implementación de los mismos de acuerdo con lo establecido por el CENIA.

2.2 Estudio preliminar

La gestión universitaria en la UCI asegura el funcionamiento de las distintas áreas de la institución educativa. Implícito en los procesos vigentes en algunas de estas áreas se realizan eventos, cuya celebración cuenta con una coordinación, planificación, evaluación y participación. Como parte del desarrollo deportivo en la universidad se realizan los Juegos Mella, en los cuales se compite en 23 deportes⁵ tanto por equipos como individuales. Son coordinados por el Departamento Docente Central de Educación Física y la Federación Estudiantil Universitaria (FEU) de la UCI, quienes organizan el evento. También se destaca en la Extensión Universitaria el acontecimiento cultural más importante: el Festival de Artistas Aficionados, donde se premian los mejores proyectos artísticos por cada facultad y a nivel de universidad en las categorías: artes plásticas, teatro, literatura, danza, audiovisual, música. Dicho festival es organizado y dirigido por la Vicerrectoría de Extensión Universitaria y la FEU de la UCI.

El vicerrector de investigación introduce la celebración de eventos investigativos internos clasificados por líneas científicas como: Jornada científica, Evento de historia de la informática, Mi web x Cuba, copas de programación, entre otros. En la UCI el proceso docente estudiantil termina con la defensa de un trabajo de diploma, siendo la Dirección de Formación del Profesional la responsable del seguimiento y control del mismo. Por su parte la Dirección de Formación Postgraduada es la responsable de definir doctorados, maestrías, especialidades y diplomados, los cuales culminan con un trabajo final. El trabajo de diploma, las tesis y tesinas tienen un flujo de trabajo que funciona como un evento.

⁵ Dato dado por el secretariado de la FEU en la UCI

Cada uno de estos eventos se gestionan según un proceso similar pero aplicando diferentes reglas de negocio. Los eventos investigativos comienzan con la constitución de un comité organizador y todos se planifican según cronogramas o programas, son coordinados por comités científicos, logísticos, académicos u organizadores, recepcionan trabajos, ponencias y proyectos que son evaluados por comisiones o tribunales constituidos al efecto y finalmente todos emiten certificados, títulos o documentos acreditativos de la participación en el evento. Todas son actividades comunes en las cuales se gestiona información de manera diferente. Se hace necesaria la realización de una solución informática que unifique dichas actividades en un proceso integrador, pero que tenga en cuenta las particularidades de la actividad en cada tipo de evento.

La elaboración de esta solución no supone gastos adicionales por concepto de salario, pues cuenta con el capital humano necesario y requiere de un esfuerzo mínimo de 2 personas y 6 meses. La solución podrá adaptarse a otro contexto que tenga un negocio similar, brindando la posibilidad de configuración de las reglas de negocio para cada actividad. Además logrará una disminución considerable del tiempo que se emplea en realizar cualquier actividad, reduciendo el tiempo de ejecución y haciendo más confiables los resultados al minimizarse el error humano. La propuesta no requiere inversión en software pues todas las herramientas que se emplearán en su desarrollo, así como las librerías empleadas, son libres y de código abierto. Por todo ello se considera que es factible el desarrollo de la aplicación y que el esfuerzo de desarrollo está plenamente justificado.

2.2.1 Modelado del negocio

Para conseguir sus objetivos, una empresa organiza su actividad por medio de un conjunto de procesos de negocio. Cada uno de ellos se caracteriza por una colección de datos que son producidos y manipulados mediante un conjunto de tareas, en las que ciertos agentes (por ejemplo trabajadores o departamentos) participan de acuerdo a un flujo de trabajo determinado. Además, estos procesos se hallan sujetos a un conjunto de reglas de negocio, que determinan la estructura de la información y las políticas de la empresa. Por tanto, la finalidad del modelado del negocio es describir cada proceso del negocio, especificando sus datos, actividades (o tareas), roles (o agentes) y reglas de negocio (26).

Debido a que en una universidad se realizan eventos de diversa índole se hizo necesario estudiar el proceso de negocio para cada tipo de evento dentro de los procesos sustantivos de la UCI, la cual se tomó como muestra. De esta forma fue posible identificar las actividades del proceso en las que se realiza gestión de información y sus reglas dentro del proceso general. Este fue modelado recogiendo las tareas comunes para cualquier evento e identificando roles comunes o genéricos que llevan a cabo estas tareas. En la **figura 3** se puede apreciar el diagrama resultado del modelado de este proceso con BPMN.



El proceso de gestión de eventos en la UCI comienza con la creación por parte del *organizador general* de un *comité organizador* mediante el listado de profesionales capacitados. Los miembros de dicho comité proceden a abrir el evento (ver **anexo 1**) publicando la convocatoria, crear y aprobar el cronograma general que regirá la planificación del mismo. Posteriormente se publican las plantillas de dicho evento. Solo en caso de que sea un evento investigativo se definen las líneas de investigación-desarrollo, se crea un banco de problemas y los posibles temas de investigación. Una vez abierto el evento, el comité organizador crea los comités organizadores de base. El *participante* inscribe el proyecto y, en caso de ser un trabajo de diploma, el comité organizador de base asigna el oponente y los tutores. El participante entrega los documentos del proyecto y el *comité organizador de base* lo aprueba o no. Para los proyectos aprobados se crean tribunales asignándoles proyectos de acuerdo a las cantidades definidas. Luego se pueden crear los cronogramas específicos o de base que definirán el momento de presentación y evaluación del proyecto (ver **anexo 2**). El participante presenta el proyecto, si el mismo posee artefactos el *tribunal* los revisa, emite y registra su opinión sobre los mismos. Si no presentan artefactos o luego de registrarse la opinión de los mismos, el tribunal emite y registra su opinión sobre la presentación. Se otorga una calificación y se emite un acta o planilla de evaluación. La presentación y evaluación termina si no se está en presencia de un trabajo de diploma, en caso de lo contrario finaliza igualmente si la nota es satisfactoria; si no es satisfactoria, el participante tiene derecho a presentar el proyecto hasta 2 veces más. Si existen más evaluaciones, se vuelve a presentar y evaluar el proyecto, sino el comité revisor archiva la información del evento y se registra la información necesaria en los certificados o títulos. Estos se imprimen y pasan a ser firmados y entregados por los comités organizadores a los participantes.

Como una entrada para la obtención y análisis de los requisitos se realizó una conceptualización inicial de la gestión de información dentro del proceso (ver **anexo 3**).



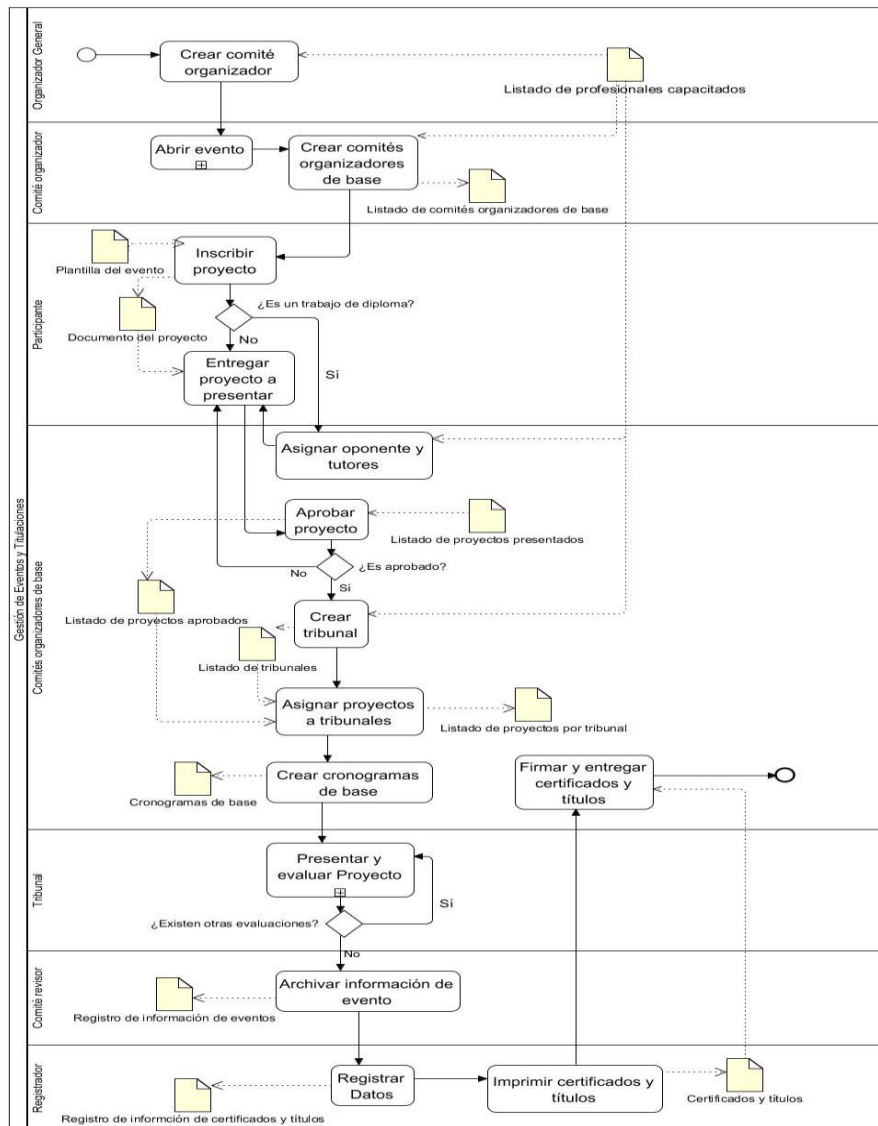


Figura 3 Modelado de Procesos del Negocio con BPMN

2.3 Obtención y análisis de los requisitos de software

2.3.1 Objeto de informatización

Del proceso vigente en el negocio de la gestión de eventos en la UCI se identificaron varias actividades en las que se realiza gestión de información y que pueden ser informatizadas:

- Gestión de comités.
- Gestión de tribunales.
- Registro de proyectos.
- Registro de Evaluaciones.



- Planificación de Cronogramas.

2.3.2 Requisitos obtenidos

Los primeros pasos de la ingeniería de requisitos apuntan a la obtención/elicitación de los mismos. En esta fase se obtienen las necesidades del cliente para posteriormente convertirlas en producto. Con el uso de las técnicas de ingeniería definidas en el capítulo 1 se obtuvo un total de 59 requisitos funcionales, de ellos 20 requisitos de prioridad muy alta, 16 de prioridad alta y 23 de prioridad media. Estos clasificados según su complejidad serían: 8 requisitos de alta, 30 de complejidad media y 21 de baja (ver artefacto *Evaluación de requisitos* del expediente de proyecto). Según sus características se clasifican en:

Requisitos del cliente:

1. Gestionar un evento teniendo en cuenta las reglas de negocio para cada tipo de evento.
2. Gestionar los cronogramas generales y de base para un evento.
3. Gestionar comités organizadores y subordinados, tribunales por comités y sus miembros, los cuales asumirán diferentes responsabilidades, teniendo en cuenta las reglas de negocio al respecto definidas para el evento.
4. Realizar la inscripción de un proyecto o trabajo en un evento y sus participantes a partir de una plantilla definida por el comité organizador.
5. Realizar la evaluación de los proyectos o trabajos que se presentan en los eventos a partir de una plantilla definida por el comité organizador.

Requisitos derivados:

1. Configurar los tipos de tribunal y tipos de comité a partir de la definición de las reglas de negocio para la asociación de miembros.
2. Gestionar los tipos de evento donde se definan las reglas del negocio para el registro de proyectos, evaluaciones y creación de sus comités organizadores.

A un bajo nivel quedaron definidos los siguientes requisitos funcionales:

Requisitos	Prioridad	Complejidad
RFET_1 Crear evento	Muy alta	Media
RFET_2 Modificar evento	Muy alta	Media
RFET_3 Mostrar eventos	Muy alta	Baja
RFET_4 Ver detalles de evento	Muy alta	Media
RFET_5 Crear proyecto a presentar	Muy alta	Alta
RFET_6 Modificar proyecto a presentar	Muy alta	Alta
RFET_7 Mostrar proyectos a presentar	Muy alta	Baja

RFET_8 Ver detalles de proyecto a presentar	Muy alta	Media
RFET_9 Asignar participantes a proyecto	Muy alta	Alta
RFET_10 Crear tribunal	Muy alta	Alta
RFET_11 Modificar tribunal	Muy alta	Alta
RFET_12 Mostrar tribunales	Muy alta	Baja
RFET_13 Ver detalles de tribunal	Muy alta	Media
RFET_14 Asignar proyectos a tribunal	Muy alta	Media
RFET_15 Crear comité	Muy alta	Alta
RFET_16 Modificar comité	Muy alta	Alta
RFET_17 Mostrar comités	Muy alta	Baja
RFET_18 Ver detalles de comité	Muy alta	Media
RFET_19 Asignar tribunales a comité	Muy alta	Media
RFET_20 Registrar evaluación de proyecto	Muy alta	Alta
RFET_21 Crear problema	Alta	Baja
RFET_22 Modificar problema	Alta	Baja
RFET_23 Mostrar problemas	Alta	Baja
RFET_24 Ver detalles de problema	Alta	Baja
RFET_25 Crear tema de investigación	Alta	Baja
RFET_26 Modificar tema de investigación	Alta	Baja
RFET_27 Mostrar temas de investigación	Alta	Baja
RFET_28 Ver detalles de tema de investigación	Alta	Baja
RFET_29 Crear cronograma general	Alta	Media
RFET_30 Modificar cronograma general	Alta	Media
RFET_31 Mostrar cronogramas	Alta	Media
RFET_32 Ver detalles cronograma general	Alta	Media
RFET_33 Crear entrada de cronograma	Alta	Media
RFET_34 Modificar entrada de cronograma	Alta	Media
RFET_35 Ver detalles entrada de cronograma	Alta	Media
RFET_36 Mostrar registro de evaluaciones	Alta	Baja
RFET_37 Mostrar informes de temas de investigación	Media	Media
RFET_38 Crear informe de tema de investigación	Media	Media
RFET_39 Revisar Informe de tema de investigación	Media	Media
RFET_40 Crear tipo de evento	Media	Media
RFET_41 Modificar tipo de evento	Media	Media
RFET_42 Mostrar tipos de evento	Media	Media



RFET_43 Ver detalles de tipo de evento	Media	Media
RFET_44 Crear tipo de tribunal	Media	Media
RFET_45 Modificar tipo de tribunal	Media	Media
RFET_46 Mostrar tipos de tribunal	Media	Media
RFET_47 Ver detalles de tipo de tribunal	Media	Media
RFET_48 Crear tipo de comité	Media	Media
RFET_49 Modificar tipo de comité	Media	Media
RFET_50 Mostrar tipos de comité	Media	Media
RFET_51 Ver detalles de tipo de comité	Media	Media
RFET_52 Crear línea de investigación-desarrollo	Media	Baja
RFET_53 Modificar línea de investigación-desarrollo	Media	Baja
RFET_54 Mostrar línea de investigación-desarrollo	Media	Baja
RFET_55 Ver detalles de línea de investigación-desarrollo	Media	Baja
RFET_56 Crear responsabilidad	Media	Baja
RFET_57 Modificar responsabilidad	Media	Baja
RFET_58 Mostrar responsabilidad	Media	Baja
RFET_59 Ver detalles responsabilidad	Media	Baja

Tabla 1 Listado de requisitos funcionales

Además se identificaron los siguientes requerimientos no funcionales:

Clasificación	Requisitos
Usabilidad	<p>RNF 1- Facilidad de uso por parte de los usuarios: el sistema debe presentar una interfaz amigable que permita la fácil interacción con el mismo y llegar de manera rápida y efectiva a la información buscada. Debe, además, ser una interfaz de manejo cómodo que posibilite a los usuarios sin experiencia una rápida adaptación.</p> <p>RNF 2- Especificación de la terminología utilizada: el sistema debe adaptarse al lenguaje y términos utilizados por los usuarios en la rama abordada con vista a una mayor comprensión por parte del cliente de la herramienta de trabajo.</p> <p>RNF 3- Potencialidades de capacitación orientadas a interfaces intuitivas, lo que enaltece la posibilidad de que el usuario aprenda mediante el uso y explotación de la herramienta.</p> <p>RNF 4- Menús: el sistema debe presentar una serie de menús tanto laterales como en barra de iconos flotantes que permitan el acceso rápido a la información por parte de los usuarios, aprovechando así las potencialidades de estas estructuras.</p>
Seguridad	<p>RNF 5- La seguridad de la base de datos está a nivel de roles, con el fin de mantener la integridad de los</p>

	<p>datos en función del acceso de cada uno de ellos, trayendo consigo además la protección de la información.</p> <p>RNF 6- Servicios Web restringidos: los servicios Web que brinde el sistema deben estar restringidos a grupos de usuarios definidos y aprobados previamente.</p> <p>RNF 7- Políticas de seguridad por usuarios y roles: el sistema debe contar con un grupo de políticas de accesibilidad a las diferentes funcionalidades del mismo en dependencia del nivel de autorización que presente un usuario determinado.</p> <p>RNF 8- Registro sistemático de incidencias: el sistema debe ser capaz de registrar el accionar del usuario, así como permitir auditorías y exámenes de las trazas tanto en tiempo real como en históricos. Se precisa un monitor de incidencia para la visualización y tratamiento de las mismas.</p>
Eficiencia	<p>RNF 9- El sistema debe soportar un tiempo de respuesta menor o igual a 5 segundos.</p> <p>RNF 10- El sistema debe soportar una conexión simultánea de más de 3000 usuarios.</p>
Soporte	<p>RNF 11- El sistema brinda como apoyo una Ayuda contextual en la cual se refleja detalladamente la explicación de cada una de las pantallas con sus respectivas funcionalidades.</p> <p>RNF 12- Referirse al documento de arquitectura: CENIA_PRE_ADASP-v1.0 (en el mismo se explica detalladamente las normas de codificación, convenciones para nomenclatura, bibliotecas de clase, el acceso y utilidades de mantenimiento.)</p>
Restricciones de diseño	<p>RNF 13- Servidor de base de datos con Postgres 8.4 o superior bajo el sistema operativo Ubuntu Server 10.4.</p> <p>RNF 14- Servidor de aplicaciones Web: Apache 2.2 o superior.</p> <p>RNF 15- Navegador Web: Internet Explorer 7 o superior, Mozilla Firefox 2.3 o superior.</p>
Requisitos para la documentación de usuarios en línea y ayuda del sistema	<p>RNF 16- Manual de usuario: el sistema deberá presentar un manual de usuario, permitiendo con ello un correcto uso de sus funcionalidades y brindarle al usuario una mayor experiencia del trabajo con el mismo.</p> <p>RNF 17- Documentación actualizada del grupo de desarrollo: se precisa que la documentación del sistema esté actualizada en todos los aspectos, fases de trabajo y ciclos de desarrollo del mismo, permitiendo con ello un respaldo tanto ingenieril como legal del desarrollo de dicho sistema.</p>
Interfaz	<p>RNF 19- Interfaz Web: la interfaz es sencilla con colores suaves a la vista y sin cúmulo de imágenes u objetos que distraigan al cliente del objetivo.</p> <p>RNF 20- Contratos: están determinados por los desarrolladores, construyendo así una vista escalable de las clases o agrupaciones de clases que permitirán un mejor encapsulamiento de las funcionalidades y una</p>

	mayor abstracción modular del sistema.
Interfaces hardware	RNF 21- La comunicación entre el servidor de aplicaciones y la base de datos se lleva a través del protocolo de conexión segura SSL. RNF 22- La comunicación entre el cliente y el servidor de aplicaciones se lleva a través del protocolo HTTPS.
Interfaces de comunicación	RNF 23- El sistema no presenta comunicaciones a otros sistemas o dispositivos como las redes de área locales.
Requisitos legales, de derecho de autor y otros	RNF 25- El sistema debe ser sometido a un análisis legal por parte de los abogados y personal autorizado con vistas a declarar su autenticidad y evitar restricciones legales para su uso y comercialización; así mismo se debe proceder a una evaluación y certificación por parte del cliente del producto.
Estándares aplicables	RNF 26- Referirse al documento de arquitectura: CENIA_PRE_ADASP-v1.0 (en el mismo se especifica los requisitos de estándares aplicables).

2.3.3 Audiencia del sistema

Las personas relacionadas con el sistema (audiencia) son todas aquellas que de una forma u otra van a interactuar con la aplicación, incluyendo a los que mantienen el sistema actualizado y en correcto funcionamiento y los que utilizan su servicio.

Agentes	Descripción
Organizador general	Se encarga de crear el comité organizador y el evento. Configura las reglas del negocio y nomencladores del evento.
Comité	Planifica el evento en la instancia correspondiente y puede crear comités subordinados y sus tribunales. Revisa y aprueba proyectos, asignándolos a tribunales. Para los eventos de investigación puede crear, líneas, problemas y temas de investigación.
Participante	Registra proyectos en un evento. En los eventos investigativos crea su informe para revisión. Puede consultar la planificación del evento.
Tribunal	Registra evaluación de los proyectos. Puede consultar la planificación del evento.
Comité revisor	Revisa y aprueba el informe de los proyectos investigativos que así lo requieran.
Administrador	Tiene acceso a todas las funcionalidades.

Tabla 2 Audiencia del sistema

2.4.1 La solución como un todo

Se pretende desarrollar una solución web genérica y configurable para gestionar eventos en la UCI, independientemente de su tipo y de las reglas de negocio que cada uno posea. Integrada además al Sistema de Gestión Universitaria en 2 niveles de funcionalidades:

- Módulo Eventos y Titulaciones⁶: donde se gestionan los eventos, comités, tribunales, cronogramas generales y sus entradas de cronograma, proyectos y sus evaluaciones. También incluirá las opciones de configuración, donde se podrán manejar los tipos de eventos, tipos de comités y tipos de tribunal, así como las responsabilidades que podrán ocupar los miembros y participantes.
- Tesis y Títulos⁷: donde se gestiona la información referente al trabajo de diploma como evento, los proyectos de trabajo de diploma y sus actas de evaluación para cada entrada de cronograma. Además de incluir la gestión del banco de problemas, temas de investigación, líneas de investigación-desarrollo y el informe de cada tema-proyecto.

La propuesta define distintos niveles de acceso a partir de roles asociados a cada una de las responsabilidades definidas para un evento, los cuales accederán mediante la vista de autenticación del SGU. Esto garantiza que cada usuario tenga acceso solo a las funcionalidades permitidas y a la información que solamente a él le concierne. Todas las vistas que presentan un gran volumen de información contienen un motor de búsqueda por criterio y filtros para garantizar rapidez en las respuestas.

Como parte del análisis se modelaron algunos diagramas para ayudar a la posterior especificación e implementación de los requisitos (ver **anexos 4, 5, 6 y 7**)

2.4 Especificación y validación de los requisitos.

2.4.1 Especificación de requisitos

Una buena especificación de requerimientos debe ser completa, consistente, modificable y traceable. La elección cuidadosa de cada una de las técnicas para la especificación influye en el éxito del producto. El resultado del estudio de estas ha conducido a una elección robustecida. Se han escogido los escenarios para la descripción del flujo de los requisitos, los prototipos para la construcción de interfaces no funcionales y el glosario para la aclaración de términos técnicos y del negocio. Se

⁶ Este módulo incluye además las funcionalidades referentes a la gestión de información en documentos acreditativos, lo cual constituye otro tema de investigación.

⁷ Este módulo incluye además las funcionalidades referentes al registro de información en los títulos de los graduados de pregrado, lo cual constituye otro tema de investigación.



obtuvieron los documentos *Especificación de requisitos* y *Glosario de términos*, los cuales pueden ser consultados en el expediente de proyecto.

A continuación se presenta un ejemplo de requerimiento especificado:

Requisito funcional Crear evento

Nº	Nombre	Descripción	Complejidad	Prioridad para cliente
RFET_1	Crear evento	<ul style="list-style-type: none"> El requisito permite crear un evento. Para crear un nuevo evento se selecciona la acción crear en la barra de íconos flotantes del listado de eventos. El Presidente del comité introduce los datos: Nombre, Tipo de evento, Descripción, Comité, Edición, Fecha de certificación podrá marcarlo como Habilitado (activo/inactivo). Si el tipo de evento tiene definida la edición por curso académico se mostrarán todos los cursos académicos posibles a seleccionar, sino se permitirá introducir un año. El Presidente del comité puede crear un nuevo evento o volver al listado de eventos. 	Media	Muy Alta

Prototipo



Crear evento:

Nombre: *
 Tipo de evento: *
 Descripción: *
 Comité: *
 Edición: *
 Fecha de certificación: *
 Habilitado
 Aceptar Cancelar

Campos	Tipos de Datos	Reglas o Restricciones
Nombre	<ul style="list-style-type: none"> Varchar 	<ul style="list-style-type: none"> Obligatorio. Único. Alfanumérico y además admite guiones, guiones bajos y espacios entre palabras. Admite entre 4 y 50 caracteres válidos. Solo admite 30 caracteres por palabra.
Tipo de evento	<ul style="list-style-type: none"> Varchar 	<ul style="list-style-type: none"> Selección. Obligatorio.
Descripción	<ul style="list-style-type: none"> Varchar 	<ul style="list-style-type: none"> Admite de 0 hasta 200 caracteres. Alfanumérico, admite guiones, guiones bajos y espacios entre palabras. Solo hasta 30 caracteres por palabra.
Habilitado	<ul style="list-style-type: none"> Boolean 	<ul style="list-style-type: none"> Selección.
Edición(curso académico)	<ul style="list-style-type: none"> Varchar 	<ul style="list-style-type: none"> Selección. Obligatorio.
Edición (año fiscal)	<ul style="list-style-type: none"> Integer 	<ul style="list-style-type: none"> Obligatorio. Admite solamente 4 caracteres. Solo números.



Comité	<ul style="list-style-type: none"> • Varchar 	<ul style="list-style-type: none"> • Selección. • Obligatorio. • Único.
Fecha de Certificación	<ul style="list-style-type: none"> • Date 	<ul style="list-style-type: none"> • Selección.
Observaciones	<ul style="list-style-type: none"> • Si se crea satisfactoriamente el evento el sistema mostrará un mensaje de confirmación <i>“El elemento ha sido creado satisfactoriamente”</i> y le permitirá al usuario crear un nuevo evento. • En caso de que el evento exista se muestra un mensaje de error: <i>“El elemento ya existe”</i>. • Si el usuario introduce caracteres extraños en el campo Nombre el sistema muestra en forma de globo un mensaje <i>“Entre solo letras, números, guiones bajos, guiones y espacios”</i>. • Si el usuario introduce caracteres extraños en el campo Edición (año fiscal) el sistema muestra en forma de globo un mensaje <i>“Entre solo números”</i>. • En caso de cancelar la acción se muestra un mensaje de advertencia <i>“¿Está seguro de realizar la acción?”</i>. • En caso que se deje un campo de los obligatorio vacío se muestra un mensaje en rojo <i>“Campo requerido”</i> encima del campo que debe ser llenado obligatoriamente. • Para todos los campos cuando llegue a la cantidad máxima de caracteres el sistema no permitirá continuar escribiendo. • En caso de que el usuario en el campo nombre introduzca menos cantidad de caracteres de los que están definidos se muestra un mensaje en rojo encima del campo <i>“Entre al menos 4 caracteres”</i>. • En caso de que el usuario en el campo Edición (año fiscal) introduzca menos cantidad de caracteres de los que están definidos se muestra un mensaje en rojo encima del campo <i>“Entre solo 4 caracteres”</i>. 	

Tabla 3 Especificación del requisito Crear evento

2.4.2 Validación de requisitos

Para la obtención de una aplicación confiable, se necesita afirmar que los requerimientos forman la solución que el cliente espera. Para ello se realiza la validación de los mismos. Los objetivos de la validación según Pressman son:



Asegurar que todos los requisitos del software se han establecido de manera precisa, que se han detectado las inconsistencias, omisiones y errores y que estos han sido corregidos.

Asegurar que los productos de trabajo cumplen con los estándares establecidos para el proceso, proyecto y producto.

Las verificaciones que incluye el proceso, se orientan a comprobar que los requerimientos descritos no tengan contradicciones, que definan las funciones propuestas por el cliente, que se puedan implementar, y que sean verificables, es decir, que sea posible realizar pruebas que demuestren que el sistema cumple con cada uno de ellos.

Para la comprobación de la validez de los requerimientos se utilizaron las técnicas: revisiones mediante listas de verificación y generación de casos de prueba.

Con el empleo de la lista de verificación propuesta por Pressman se validaron los requisitos en 2 iteraciones en vistas de probar la comprensibilidad, rastreabilidad y adaptabilidad de cada uno (ver **anexo 10**). En la primera se aprobaron 44 requisitos y se rechazaron 15, los cuales se corrigieron y aceptaron luego en la segunda iteración, como se puede ver en la figura 4.

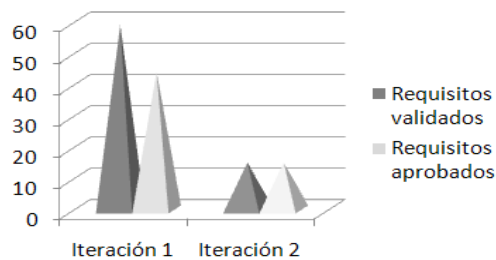


Figura 4 Resultado de validación con lista de verificación.

Se destinaron también diseños de casos de prueba (DCP) para asegurar que cada uno de los requerimientos es verificable y que su futura implementación podrá realizarse sin demasiadas dificultades.

Además se aplicaron criterios para validar los requisitos del cliente y del producto propuestos por el CENIA, obteniéndose el 100% de requisitos aprobados.

2.5 Descripción de la arquitectura de sistema

Una arquitectura de software de un programa o sistema de computación es la estructura o estructuras del sistema, el cual comprende componentes, las propiedades visibles externas de dichos componentes y las relaciones entre ellos (27).



2.5.1 Vista de presentación

La arquitectura de la información es un proceso iterativo, transversal, que se da a lo largo de todo el diseño del sitio y en cada una de sus fases, para asegurarse de que los objetivos de su producción y del desarrollo de la interfaz se cumplen de manera efectiva (28).

Se puede apreciar en el **anexo 11** el mapa de navegación que describe la estructura del contenido en la solución descrita.

La construcción de los prototipos de interfaz se realizó según los documentos de arquitectura de información definidos para el SGU.

2.5.2 Vista de integración

Un patrón de arquitectura es una descripción de elementos y los tipos de relación, junto con un grupo de restricciones de cómo deben ser usados (29).

Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos. El patrón de llamada y retorno MVC, se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página; el modelo es el sistema de gestión de base de datos y la lógica de negocio; el controlador es el responsable de recibir los eventos de entrada desde la vista (30). En la solución actual el patrón MVC, como se muestra en la figura 6, está implícito en el marco de trabajo GUUD.

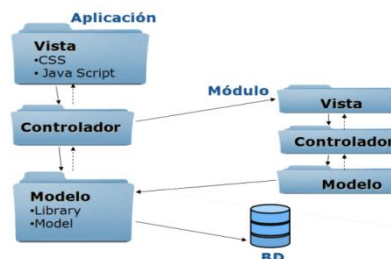


Figura 5 Patrón arquitectónico Modelo-Vista-Controlador implementado en GUUD.

Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias (31).

En el diseño de la solución propuesta se tuvieron en cuenta los patrones de asignación de responsabilidades (GRASP) y los patrones GOF (Gang-Of-Four) que se verán a continuación:

GRASP:



Los patrones GRASP (General Responsibility Assignment Software Patterns, Patrones Generales de Software para Asignar Responsabilidades) Para el diseño del módulo se tuvieron en cuenta los 5 patrones, ya que el marco de trabajo usado busca un máximo rendimiento y flexibilidad en sus soluciones y pone en práctica estos patrones para lograr un sistema reusable y flexible.

Experto: el patrón experto en información define el principio básico de asignación de responsabilidades. Indica que la responsabilidad de la creación de un objeto debe recaer sobre la clase que conoce toda la información necesaria para crearlo. Se evidencia en las clases librerías, que son las que cuentan con la información necesaria para cumplir la responsabilidad es sobre los elementos de negocio.

Creador: permite identificar quién debe ser el responsable de la instanciación de nuevos objetos o clases. Se utilizó para identificar qué clase A debe crear elementos de una clase B, apoyándose en que la clase A debería: contener, agregar, registrar, utilizar y tener los datos de inicialización de la clase B. En la clase loader que es el objeto load de las clases controladoras, encarga de cargar los elementos del marco de trabajo dígame, librerías, modelos.

Alta cohesión: este patrón define que la información que almacena una clase debe de ser coherente y está en mayor medida relacionada con la clase. En el sistema propuesto es necesario controlar la complejidad de cada clase utilizada para mantener un buen comportamiento de las mismas, por esto, las clases que fueron identificadas con una gran cantidad de funcionalidades se dividieron en otras clases de manera que se repartiera equitativamente el peso de la complejidad, manteniendo además la coherencia de las clases.

Bajo acoplamiento: este patrón se utilizó con la idea de tener las clases lo menos ligadas posible. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

Controlador: el patrón controlador se utilizó para que sirviera como intermediario entre cada una de las capas, de forma tal que se garantice la comunicación entre los eventos externos del sistema, las clases controladoras se encargan de obtener datos, enviarlos las librerías y las vistas.

GOF:

Los *patrones de creación* abstraen la forma en la que se crean los objetos, permitiendo tratar las clases a crear de forma genérica dejando para más tarde la decisión de qué clases crear o cómo crearlas.



Abstract Factory (Fábrica abstracta): permite trabajar con objetos de distintas familias de manera que las familias no se mezclen entre sí y haciendo transparente el tipo de familia concreta que se esté usando.

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

Singleton (Instancia única): garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Todas las clases controladoras, son instancias únicas.

Los *patrones de comportamiento* estudian las relaciones entre llamadas entre los diferentes objetos, normalmente ligados con la dimensión temporal.

Mediator: (Mediador): define un objeto que coordine la comunicación entre objetos de distintas clases, pero que funcionan como un conjunto. Las librerías que funcionan como mediadoras entre las clases controladoras y las modelos o acceso a datos.

Observer: (Observador): define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él. En la clase loader que es el objeto load de las clases controladoras, se cargan los elementos del marco de trabajo, dígame librerías, modelos y se encarga de actualizar la controladora instanciada.

2.5.3 Vista de datos

Un modelo de datos es un conjunto de conceptos que sirven para describir la estructura de una base de datos: los datos, las relaciones entre ellos y las restricciones que deben cumplirse sobre los mismos. Los modelos de datos contienen también un conjunto de operaciones básicas para la realización de consultas y actualizaciones de datos. Además los modelos de datos más modernos incluyen conceptos para especificar comportamiento permitiendo especificar un conjunto de operaciones definidas por el usuario (32).

En el **anexo 12** se muestra el modelo físico de la vista de datos.

2.5.4 Vista de despliegue

La Arquitectura Cliente-Servidor consiste básicamente en un cliente que realiza peticiones a otro programa (el servidor) que le da respuesta. Aunque esta idea se puede aplicar a programas que se ejecutan sobre una sola computadora es más ventajosa en un sistema operativo multiusuario y distribuido a través de una red de computadoras (33).



En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema (33). La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es un sólo programa. Los tipos específicos de servidores incluyen los servidores web, los servidores de archivo, los servidores del correo, etc. Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma. El SGU tiene su propio diagrama de despliegue basado en este estilo. Debido a que la solución propuesta estará integrada en este sistema la misma retoma este diseño (**figura 7**). En la PC cliente estará instalada la aplicación, esta se conectará con el servidor de web mediante una conexión segura HTTPS, este a su vez se conecta con el servidor de base datos a través del protocolo TCP/IP.

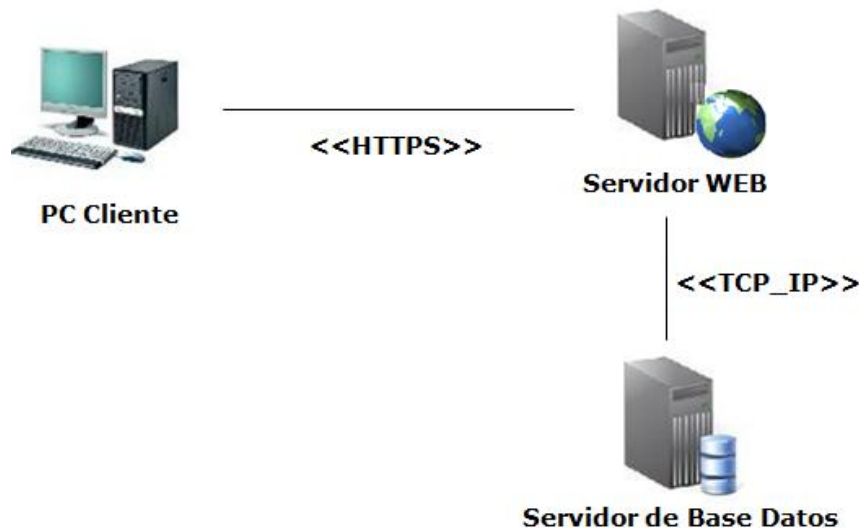


Figura 6 Diagrama de despliegue del sistema

2.6 Conclusiones

El conocimiento íntegro del dominio del negocio y de los procesos vigentes en el mismo, es la base para la eficiencia de un estudio preliminar necesario en la producción de cualquier software. En este capítulo se realizó una evaluación que demostró la factibilidad del desarrollo de la solución. Se modeló el proceso del negocio a partir del cual se identificaron las actividades en las que se realiza gestión de información quedando definida la audiencia del sistema. Esto condujo a la obtención de requisitos a diferentes niveles. Finalmente se concibió la idea de una aplicación web genérica que aplica los estándares y vistas de arquitectura definidas por el CENIA para el SGU.



Capítulo 3. Implementación y validación de la solución propuesta

3.1 Introducción

Existen disímiles técnicas y estilos de programación, pero las tendencias actuales apuntan a la utilización de las más asequibles y eficientes, incluyendo el empleo de los estándares y estructuras de control establecidos por el Centro de Informatización Universitaria. Este capítulo describe la implementación de la solución propuesta en el capítulo anterior. Se define además una estrategia de pruebas con el fin de validar la solución implementada y comprobar su correcto funcionamiento.

3.2 Técnicas de programación

Las técnicas de programación constituyen una parte fundamental en el proceso de desarrollo e ingeniería del software dentro del ámbito informático (34). Cada técnica tiene sus propias características y distintos métodos para darle solución a determinados problemas, es de gran importancia aprender a implementarlas a la hora de realizar cualquier proyecto de desarrollo de software.

3.2.1 Programación modular

La programación modular es una técnica donde los procedimientos con una funcionalidad común son agrupados en diferentes módulos. Cada módulo contendrá sus propios datos, lo que permitirá que cada cual maneje un estado interno que será modificado por las llamadas a procedimientos de estos. Existe un estado para cada módulo y estos existen aunque sea una vez en todo el programa. Por lo que un programa ya no consistirá en una sección, sino que estará dividido en varias secciones más pequeñas que interactúan a través de llamadas a procedimientos y que integran el programa en su totalidad (34). El programa principal coordina las llamadas a procedimientos en módulos separados y pasa los datos apropiados en forma de parámetros.

3.2.2 Programación orientada a objetos (POO)

En la POO se debe tener tantos objetos-lista como sea necesario. En esta programación se manda un mensaje directamente al objeto-lista en cuestión. Cada objeto implementa su propio módulo, permitiendo por ejemplo que coexistan muchas listas. Cada objeto es responsable de inicializarse y destruirse en forma correcta. No existe la necesidad de llamar explícitamente al procedimiento de creación o de terminación (35). La POO está basada en cuatro aspectos: definición de tipos de datos abstractos, herencia, encapsulamiento y polimorfismo.

En la POO encapsular significa que se reúne y controla todo el grupo resultante en un conjunto y no de forma individual. La abstracción es un término externo al objeto, que controla la forma en que es visto por los demás. La herencia se define como una jerarquía de clases derivadas y la relación entre estas, donde se comparte la estructura y el comportamiento de una o más clases consideradas como clases



padres. El polimorfismo constituye la definición de múltiples clases con funcionalidades diferentes, pero con métodos o propiedades denominados de forma idéntica (35).

3.3 Estilos de programación

La motivación que tienen los estilos de programación está relacionada con la comunicación entre personas que facilitan los lenguajes de programación. Las reglas de los estilos son muy flexibles, lo cual no significa que se va a estar cambiando el estilo en cada momento, todo lo contrario, es importante mantener las reglas rígidas dentro de un proyecto. No basta con escribir un programa que funcione, el código tiene que estar bien escrito para que se construya un buen programa (30). Existen diversas cualidades que se ven beneficiadas por el uso de un buen estilo de programación entre las cuales se encuentran:

Extensibilidad: la facilidad con que se adapta el software a cambios de especificación. Un buen estilo de código fomenta programas que no solo resuelven el problema, sino que también reflejan claramente la relación problema/solución. Esto tiene como efecto que muchos cambios simples en el problema reflejen de forma obvia los cambios a hacer en el programa.

Verificabilidad: la facilidad con que pueden comprobarse propiedades de un sistema. Si el estilo de código hace obvia la estructura del programa, eso ayuda a verificar que el comportamiento sea el esperado.

Reparabilidad: la posibilidad de corregir errores sin demasiado esfuerzo.

Capacidad de evolución: la capacidad de adaptarse a nuevas necesidades.

Comprensibilidad: la facilidad con que el programa puede ser comprendido.

Existen diferentes estilos de programación como son: Estilo K&R, Estilo Allman, Estilo BSD KNF, Estilo Whitesmiths y Estilo GNU

3.4 Estándares de codificación

Un estándar de codificación por lo general son reglas que se siguen para la escritura del código fuente, de tal manera que a otros programadores se les facilite entender el código.

3.4.1 Identación, llaves de apertura y cierre y tamaño de líneas

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

3.4.2 Conversión de nomenclatura

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



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

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

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

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

3.4.3 Estructuras de control

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

3.4.4 Documentación

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

```
/**
 * Description of investigacion
 * PHP5
 * @category Libreria
 * @package eventos_titulaciones
 * @author ernesto
 */
```

Figura 7 Documentación asociada a las clases

3.5 Integración con el Sistema de gestión universitaria.

La solución se integra al Sistema de Gestión Universitaria a través de 2 niveles:

1. Módulo genérico dentro del núcleo del Sistema.
2. Módulo adaptado a los subsistemas de Pregrado y Postgrado

1er nivel



En el subsistema núcleo se encuentra el módulo Eventos y Titulaciones, donde se realizan las funcionalidades genéricas de gestión y configuración de eventos, y que se integra con los otros módulos del núcleo para obtener la información de las personas, las estructuras y garantizar la seguridad de la solución. Más específicamente, Eventos y Titulaciones:

- Adquiere la información de las personas del módulo Gestión de personal, implícita en el registro de proyectos, y la gestión de tribunales y comités.
- Obtiene las estructuras del módulo Estructura y Composición por las que se clasificarán los comités.
- Se relaciona con el módulo Seguridad para limitar el acceso a las funcionalidades a los usuarios del sistema, y para asignar un rol a cada responsabilidad de los miembros de comités al crearse los mismos.

2do nivel

Están presentes los subsistemas de Gestión Académica de Pregrado y Postgrado donde se encuentran los módulos Tesis y Títulos y Titulaciones respectivamente. En estos módulos se realizan las funcionalidades específicas para la gestión de tesis, trabajos de diploma, certificados y títulos para futuros egresados. Los temas de investigación gestionados en Tesis y Títulos se incluyen como dato variable en la gestión de proyectos en Eventos y Titulaciones.

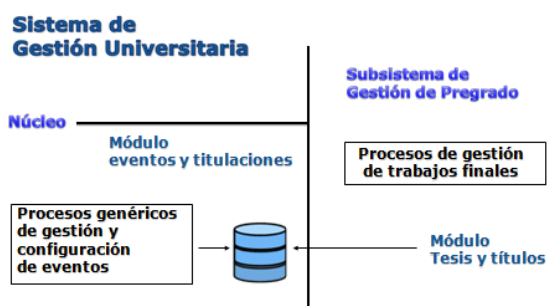


Figura 8 Diagrama de integración

3.6 Interfaces propuestas

El módulo Evento y Titulaciones implica la gestión de eventos, de proyectos, tribunales, comités y cronogramas, entre otras; dichas funcionalidades manejan datos variables y a partir de ellas se administran las demás funcionalidades del sistema.

La aplicación incluye como actividad primordial la creación de eventos. Ello implica la introducción de los datos de un evento, así como la clasificación de algunas de sus cualidades mediante datos predefinidos.



Figura 9 Interfaz de Crear evento.

La planificación de cronogramas a partir de la utilización de árboles visuales permite al usuario realizar este proceso de forma fácil, organizada e intuitiva.

Figura 10 Interfaz de Mostrar cronogramas generales.

La creación de comités como estructuras coordinadoras de los eventos se realizan teniendo en cuenta las reglas definidas en el tipo de comité que se desea crear. Se puede realizar una búsqueda por criterios y filtros seleccionando los miembros del comité y asignándoles una responsabilidad.

Figura 11 Interfaz de Crear comité



3.7 Pruebas

La construcción de un sistema software tiene como objetivo satisfacer una necesidad planteada por un cliente. Pero ¿cómo se puede saber si el producto construido se corresponde exactamente con lo que el cliente les pidió y funciona correctamente? Por ese motivo es que se hace necesario llevar a cabo, en paralelo al proceso de desarrollo, un proceso de evaluación o comprobación de los distintos productos o modelos que se van generando (36). Para dicho proceso se pueden usar distintos tipos de técnicas. En general, estas técnicas se agrupan en dos categorías: Técnicas de Evaluación Estáticas y Técnicas de Evaluación Dinámicas o pruebas. Las pruebas de software son los procesos que permiten verificar y revelar la calidad de un producto software. Son utilizadas para identificar posibles fallos de implementación, calidad o usabilidad de un programa. Básicamente es una fase en el desarrollo de software consistente en probar las aplicaciones construidas. Las pruebas de software se integran dentro de las diferentes fases del ciclo de vida del proyecto dentro de la Ingeniería de software. Así se ejecuta un programa y mediante técnicas experimentales se trata de descubrir que errores tienen. Para comprobar que la aplicación cumple con los requisitos funcionales definidos, es necesario realizarle pruebas antes de dar por terminado su proceso de implementación. El propósito de estas pruebas es simular una carga de producción real y observar cómo se comporta el sistema bajo cargas intensivas. Esto permite solucionar los problemas de rendimiento, antes de poner la aplicación en marcha (36).

3.7.1 Niveles y técnicas de prueba

El proceso de evaluación de un software debe permitir comenzar por los componentes más simples y más pequeños e ir avanzando progresivamente hasta probar todo el software en su conjunto (36). Para ello se cuenta con cuatro pasos fundamentales que se implementan de manera secuencial, los mismos son representados en la figura.



Figura 12 Niveles de prueba

Pruebas unitarias

La prueba de unidad se concentra en el esfuerzo de verificación de la unidad más pequeña: el componente o módulo de software. Tomando como guía la descripción del diseño al nivel de componentes, se prueban importantes caminos de control para descubrir errores dentro de los límites



del módulo. El alcance restringido que se ha determinado para las pruebas de unidad limita la relativa complejidad de las pruebas y los errores que estas descubren. Las pruebas de unidad se concentran en la lógica del procesamiento interno y en las estructuras de datos dentro de los límites de un componente. Se puede aplicar en paralelo a varios componentes (37).

Pruebas de integración

Aun cuando los módulos de un programa funcionen bien por separado es necesario probarlos conjuntamente. Para ello se aplican las pruebas de integración, el objetivo de las mismas es verificar el software ensamblando todos los módulos probados previamente.

La integración puede ser de manera no incremental, es decir, a combinar todos los módulos y probar todo el programa en su conjunto; o incremental, en la que el programa se prueba en pequeñas porciones posibilitando una mejor detección de fallos (36).

Además, las pruebas de integración incremental pueden realizarse a través de estrategias como (9):

- Ascendente: en la que los módulos se integran al descender por la jerarquía de control, empezando con el módulo de control principal.
- Descendente: empieza la prueba con los componentes de los niveles más bajos de la estructura del programa.

Pruebas del sistema

Este tipo de pruebas tiene como propósito ejercitar profundamente el sistema para verificar que se han integrado adecuadamente todos los elementos del sistema (hardware, otro software, etc.) y que realizan las funciones adecuadas. Concretamente se debe comprobar que:

- Se cumplen los requisitos funcionales establecidos.
- El funcionamiento y rendimiento de las interfaces hardware, software y de usuario.
- La adecuación de la documentación de usuario (36).

3.7.2 Métodos y técnicas de prueba

Para la evaluación dinámica del software se usan métodos como:

– *Método de caja blanca o estructural*: se basan en un minucioso examen de los detalles procedimentales del código a evaluar, por lo que es necesario conocer la lógica del programa.

Dicho método incluye una serie de técnicas que se aplican atendiendo al comportamiento interno y a la estructura del programa, las mismas se clasifican en: cobertura de sentencias, cobertura de decisión, cobertura de condiciones, cobertura decisión/condición, cobertura de condición múltiple y cobertura de caminos. En esta última se escriben casos de prueba suficientes para que se ejecuten todos los caminos de un programa. Entendiendo camino como una secuencia de sentencias encadenadas desde la entrada del programa hasta su salida.



– *Método de caja negra o funcional*: realiza pruebas sobre la interfaz del programa a probar, donde la interfaz son las entradas y salidas de dicho programa, haciéndose innecesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar.

Además, como parte del mismo se aplican técnicas como: análisis de valores límite, métodos basados en grafos, pruebas de comparación, análisis causa-efecto y particiones de equivalencia, en la misma divide el campo de entrada de un programa en *clases de datos* de los que se pueden derivar casos de prueba (36).

3.7.3 Estrategia de pruebas.

La estrategia proporciona un mapa que describe los pasos que se darán como parte de la prueba. Por tanto, cualquier estrategia de pruebas debe incorporar la planeación de las pruebas, el diseño de casos de pruebas, la ejecución de las pruebas y la recolección y evaluación de datos resultantes. Además de integrar métodos de diseño de casos de prueba en una serie bien planeada de pasos que desembocará en la eficaz construcción del software (10).

La estrategia propuesta recorre los tres niveles de prueba anteriormente descritos.

Primeramente se realizarán pruebas unitarias con el objetivo de corroborar la funcionalidad de los componentes de la solución de forma independiente, aplicando el método de caja blanca con la técnica cobertura de caminos. La misma se basa en la ejecución de las sentencias de la solución y la comprobación de sus condiciones.

3.8 Aplicación y resultados de las pruebas.

3.8.1 Pruebas de caja blanca

Para efectuar las pruebas de caja blanca a un algoritmo primeramente se debe realizar el análisis de la complejidad ciclomática del mismo, pues este valor define la cantidad máxima de caminos independientes del programa y el límite superior para el número de casos de prueba que se deben llevar a cabo, para asegurar que se ejecute al menos una vez cada sentencia y cada condición.

Para realizar los casos de pruebas, es necesario cumplir con las siguientes exigencias:

Descripción: se hace la entrada de datos necesaria, validando que ningún parámetro obligatorio pase nulo al procedimiento o no se entre algún dato erróneo.

Condición de ejecución: se especifica cada parámetro para que cumpla una condición deseada y poder ver el funcionamiento del procedimiento.

Entrada: se muestran los parámetros que entran al procedimiento.

Resultados esperados: se expone el resultado que se espera que devuelva el procedimiento.

Ejemplo de código probado:

Modificar comité



A partir del código que se encuentra en el **anexo 13** se construye el grafo de flujo asociado.

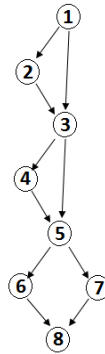


Figura 13 Grafo flujo modificar

Aplicación de las fórmulas para calcular la complejidad ciclomática. Para que el resultado sea correcto, debe ser el mismo para las tres fórmulas.

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

2. $V(G) = NP + 1$.

3. $V(G) = R$.

Donde: $\left\{ \begin{array}{l} A: \text{aristas.} \\ R: \text{regiones.} \\ N: \text{nodos. NP: nodos predicados.} \end{array} \right.$

Sustituyendo los valores se obtiene que:

1. $V(G) = (10-8)+2 = 4$

2. $V(G) = 3+1 = 4$

3. $V(G) = 4$

El cálculo efectuado mediante las tres fórmulas ha dado el mismo valor, por lo que se puede decir que la complejidad ciclomática del código es de dos, lo que significa que existen dos posibles caminos por donde el flujo puede circular, este valor representa el límite mínimo del número total de casos de pruebas para el procedimiento tratado.

Representación de los caminos básicos por los que puede recorrer el flujo:

Camino básico #1: 1-2-3-4-5-6-8

Camino básico #2: 1-2-3-4-5-7-8

Camino básico #2: 1-3-4-5-6-8

Camino básico #2: 1-2-3-5-6-8

Este algoritmo está conformado por varias condicionales que se ejecuta, de esta manera, solo es necesario probar si las condicionales se ejecutan completamente, garantizando una cobertura total.

Caso de prueba para el camino básico #1. Para el camino uno se conformó el siguiente caso de prueba:



Descripción: a la hora de modificar el comité hay que verificar que venga en el arreglo el id de comité, después se verifica que se obtenga el comité, y si el estado es activo se modifica correctamente.

Condición de ejecución: los datos de entrada son correctos, y se modificó correctamente

Entrada: se modifican los datos de la base de datos con los nuevos entrado en la aplicación. Ejemplo
`$this->template->set_data('id_comite', $comite->id_comite)`

Resultados esperados: se espera que el método modifique los datos correctamente.

Caso de prueba para el camino básico #2. Para el camino dos se conformó el siguiente caso de prueba:

Descripción: a la hora de modificar el comité hay que verificar que venga en el arreglo el id de comité, después se verifica que se obtenga el comité, y si el estado es inactivo se modifica correctamente.

Condición de ejecución: los datos de entrada son correctos, y se modificó correctamente

Entrada: se modifican los datos de la base de datos con los nuevos entrado en la aplicación. Ejemplo
`$this->template->set_data('id_comite', $comite->id_comite)`

Resultados esperados: se expone el resultado que se espera que devuelva el procedimiento.

Caso de prueba para el camino básico #3. Para el camino tres se conformó el siguiente caso de prueba:

Descripción: a la hora de modificar el comité hay que verificar que se obtenga el comité, y si el estado es activo se modifica correctamente.

Condición de ejecución: los datos de entrada son correctos, y se modificó correctamente

Entrada: se modifican los datos de la base de datos con los nuevos entrado en la aplicación. Ejemplo
`$this->template->set_data('id_comite', $comite->id_comite)`

Resultados esperados: se expone el resultado que se espera que devuelva el procedimiento.

Caso de prueba para el camino básico #4. Para el camino cuatro se conformó el siguiente caso de prueba:

Descripción: a la hora de modificar el comité hay que verificar que venga en el arreglo el id de comité, y si el estado es activo se modifica correctamente.

Condición de ejecución: los datos de entrada son correctos, y se modificó correctamente

Entrada: se modifican los datos de la base de datos con los nuevos entrado en la aplicación. Ejemplo
`$this->template->set_data('id_comite', $comite->id_comite)`

Resultados esperados: se expone el resultado que se espera que devuelva el procedimiento.

Modificar tribunal

Seguidamente se construye el grafo de flujo asociado al código que aparece en el **anexo 14**.



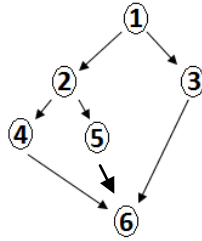


Figura 14 Grafo flujo modificar tribunal

Aplicación de las fórmulas para calcular la complejidad ciclomática. Para que el resultado sea correcto, debe ser el mismo para las tres fórmulas.

$$1. V(G) = (A - N) + 2.$$

$$2. V(G) = NP + 1.$$

$$3. V(G) = R.$$

Donde: $\left\{ \begin{array}{l} A: \text{aristas.} \\ R: \text{regiones.} \\ N: \text{nodos. NP: nodos predicados.} \end{array} \right.$

Sustituyendo los valores se obtiene que:

$$1. V(G) = (7-6)+2 = 3$$

$$2. V(G) = 2+1 = 3$$

$$3. V(G) = 3$$

El cálculo efectuado mediante las tres fórmulas ha dado el mismo valor, por lo que se puede decir que la complejidad ciclomática del código es de tres, lo que significa que existen tres posibles caminos por donde el flujo puede circular, este valor representa el límite mínimo del número total de casos de pruebas para el procedimiento tratado.

Representación de los caminos básicos por los que puede recorrer el flujo:

Camino básico #1: 1-2-4-6

Camino básico #2: 1-2-5-6

Camino básico #3: 1-3-6

Este algoritmo está conformado por varias condicionales que se ejecutan, de esta manera, solo es necesario probar si las condicionales se ejecutan completamente, garantizando una cobertura total.

Caso de prueba para el camino básico #1. Para el camino uno se conformó el siguiente caso de prueba:

Descripción: para modificar un tribunal se debe verificar que en el arreglo venga el nombre del tribunal a modificar, si se modifica correctamente le muestra un mensaje al usuario.

Condición de ejecución: los datos de entrada son correctos, y se modificó correctamente



Entrada: es un arreglo que contiene todos los datos necesarios para modificar los tribunales. Ejemplo `$datos_finales= array(nombre_tribunal->"pepe", descripción->"", estado->"activo");`

Resultados esperados: se espera que el método modifique los datos en la base datos.

Caso de prueba para el camino básico #2. Para el camino dos se conformó el siguiente caso de prueba:

Descripción: para modificar un tribunal se debe verificar que en el arreglo venga el nombre del tribunal a modificar, si no se modifica correctamente le muestra un error al usuario.

Condición de ejecución: los datos de entrada son correctos, ocurrió un error al modificar.

Entrada: es un arreglo que contiene todos los datos necesarios para modificar los tribunales. Ejemplo `$datos_finales= array(nombre_tribunal->"pepe", descripción->"", estado->"activo");`

Resultados esperados: ocurre un error a la hora de modificar el tribunal.

Caso de prueba para el camino básico #3. Para el camino tres se conformó el siguiente caso de prueba:

Descripción: son incorrectos los datos de entrada por lo que se muestra un error.

Condición de ejecución: los datos de entrada no son correctos.

Entrada: un arreglo vacío. Ejemplo `$datos_finales= array();`

Resultados esperados: se muestre un error

3.8.2 Pruebas de integración

Además se examinó la integración con el objetivo de verificar que los componentes de la propuesta se combinan de manera satisfactoria. La relación entre cada uno de ellos se representó de la siguiente manera:



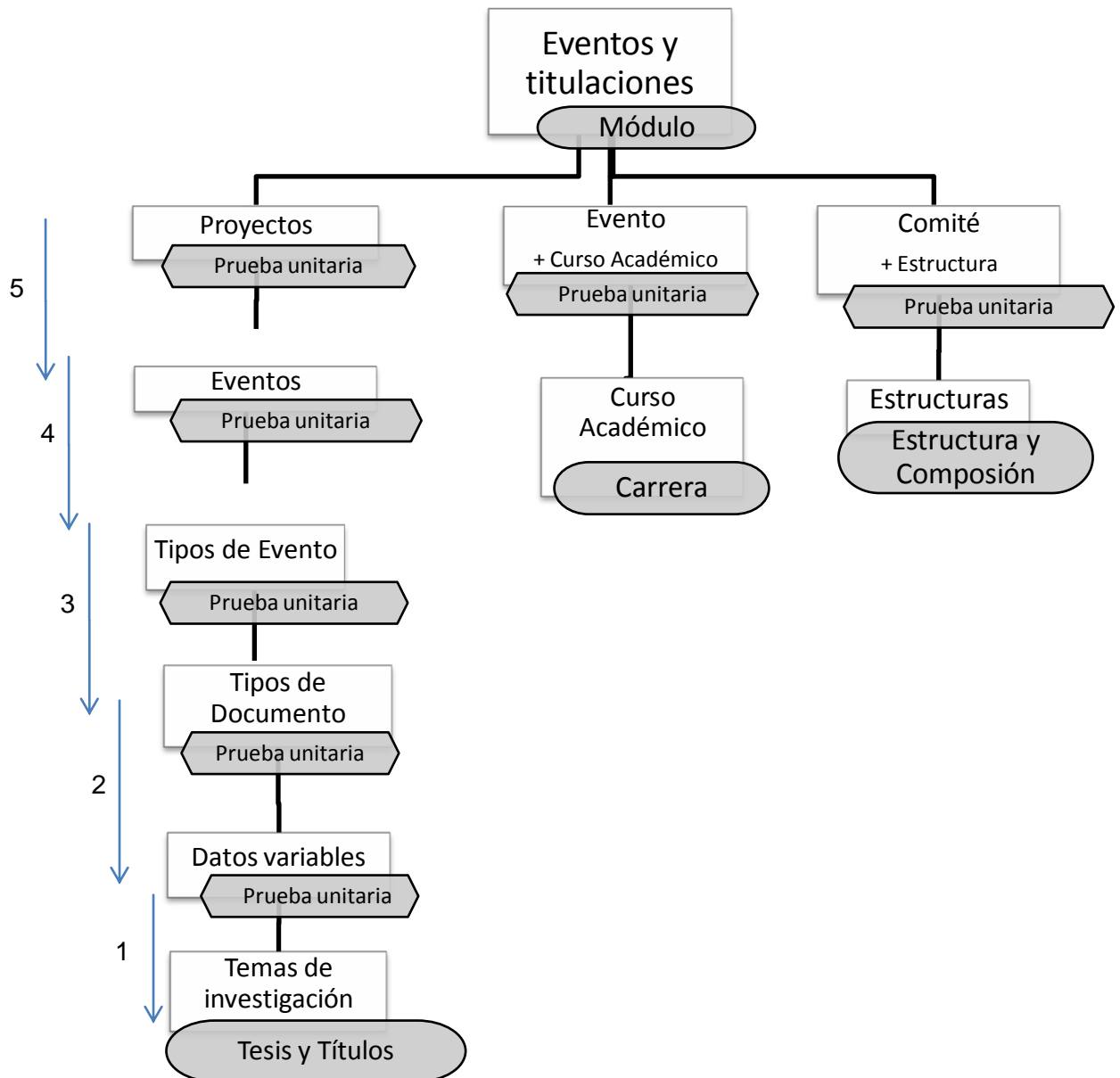


Figura 15 Diagrama de componentes a probar con integración ascendente

A partir de la gráfica, se identificaron los elementos de los componentes que hacen posible la integración, y se procedió a chequear la interconexión de los mismos de par en par mediante pruebas de integración incremental ascendente.



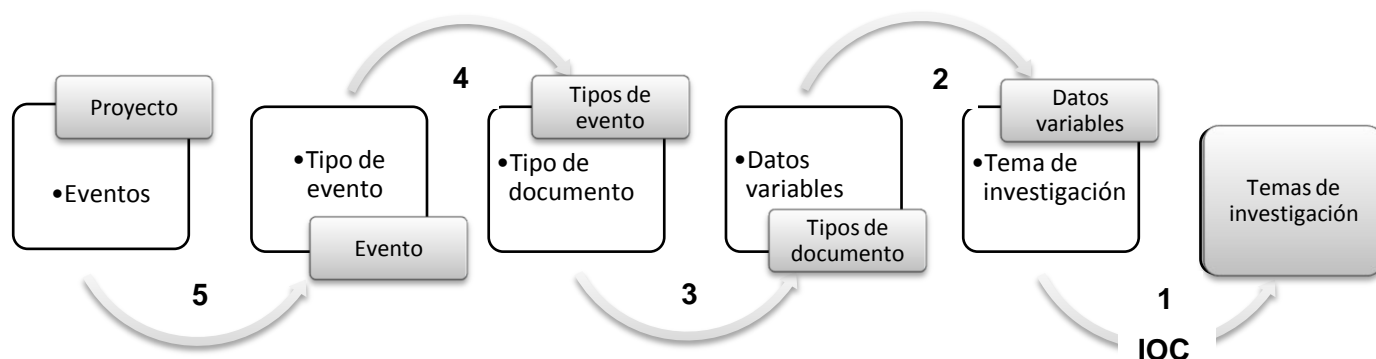


Figura 16 : Diagrama relación en la integración de proyecto y temas de investigación

Diseño de caso de prueba de integración: una vez realizadas las pruebas de unidad, se verifican los grupos integrados representados anteriormente, concentrándose en validar las funciones que se relacionan.(ver los restantes en **anexo 15**)

Condiciones de entrada

Elemento	Estados
Base de datos(BD)	activa, con problemas.
Eventos	ausente, presente, vacío.
Proyecto	activo, con problemas.

Caso de Prueba 5				
Componente que se integra: Proyecto				
Componente a integrar: Eventos				
Condiciones de ejecución: El módulo Eventos y titulaciones debe tener introducidos los datos en la base de datos y debe existir conexión con la misma.				
Objetivo de la prueba: Verificar que se puede almacenar el Evento seleccionado como información del proyecto.				
Escenarios	Entradas	Condiciones de entrada	Salidas	Descripción
Se muestran eventos correctamente.		BD activa, Eventos presente, Proyecto activo	El sistema muestra el listado con los eventos existentes.	El sistema obtiene los eventos mostrándolos en un listado, con la opción de escoger uno de ellos.
No se muestran los eventos.		BD con problemas, eventos presente, proyecto activo.	El sistema muestra un mensaje de error.	El sistema no muestra los eventos existentes.
		BD activa, Eventos ausente, Proyecto activo.		
		BD activa, Eventos ausente, Proyecto con		



		problemas.		
		BD activa, Eventos vacío, Proyecto activo.	El sistema muestra un listado vacío.	El sistema no muestra eventos.
Se guardan los datos del evento como información del proyecto.	Se selecciona el evento del listado.	BD activa, Eventos presente, Proyecto activo.	El sistema almacena el identificador del evento seleccionado como dato de la tabla correspondiente al proyecto.	Una vez mostrado el listado de eventos, el usuario selecciona el evento y presiona el botón aceptar. El sistema almacena el dato como información del proyecto.
No se guarda el evento seleccionado como dato del proyecto.		BD con problemas, Eventos presente, Proyecto activo.	El sistema muestra un mensaje de error	Una vez seleccionado el evento el usuario escoge la opción aceptar. El sistema no guarda el evento como dato del proyecto, muestra un mensaje de error.
		BD activa, Eventos presente, Proyecto con problemas.		

Condiciones de entrada

Elemento	Estados
Base de datos(BD)	Activa, con problemas.
Tipos de evento	Ausente, presente, vacío.
Eventos	Con problemas, activo.

Caso de Prueba 4
Componente que se integra: Eventos
Componente a integrar: Tipos de evento
Condiciones de ejecución: El módulo Eventos y titulaciones debe tener introducidos los datos en la base de datos y debe existir conexión con la misma.
Objetivo de la prueba: Verificar que al crear y modificar un evento se pueda guardar como dato del mismo el Tipo de evento.



Escenarios	Entradas	Condiciones de entrada	Salidas	Descripción
Se muestran los Tipos de evento correctamente.		BD activa, Tipos de evento presente, Eventos activo.	El sistema muestra en un campo de selección los nombres de los Tipos de eventos existentes.	El sistema obtiene los Tipos de eventos mostrándolos en campo de selección.
No se muestran los Tipos de evento.		BD con problemas, Tipos de eventos presente, Eventos activo.	El sistema muestra un mensaje de error.	El sistema no muestra los Tipos de evento existentes.
		BD activa, Tipos de evento ausente, Eventos activo.		
		BD activa, Tipos de evento presente, Eventos con problemas.		
		BD activa, Tipos de evento vacío, Eventos activo.	El sistema muestra un listado vacío.	El sistema al no obtener ningún dato muestra un listado vacío.
Se guardan el tipo de evento como información del evento.	Se selecciona el tipo de evento.	BD activa, Tipos de evento presente, Eventos activo.	El sistema almacena el identificador del tipo de evento seleccionado como dato de la tabla correspondiente al evento.	El usuario selecciona el tipo de evento. El sistema almacena el dato como información de evento en la tabla correspondiente de la base de datos.
No se guarda el tipo de evento seleccionado como dato del evento.	Se selecciona el tipo de evento.	BD con problemas, Tipos de evento presente, Eventos activo.	El sistema muestra un mensaje de error.	Una vez seleccionado el tipo de evento, el usuario escoge la opción aceptar. El sistema no guarda el dato, muestra un mensaje de error.



Diseño de caso de prueba del sistema: mediante ellos se valida el desempeño general de la solución a través de condiciones de entradas, mediante las cuales se obtiene una determinada respuesta del sistema.

3.8.3 Pruebas de sistema

Diseño de caso de prueba del sistema: mediante ellos se valida el desempeño general de la solución a través de condiciones de entradas, mediante las cuales se obtiene una determinada respuesta del sistema.

Finalmente las pruebas al sistema persiguieron la meta de verificar el funcionamiento de la solución. Se propuso para ambos niveles el método de caja negra a través de la técnica particiones equivalentes, diseñando casos de prueba (DCP). Todos los DCP pueden consultarse en el expediente de proyecto.

3.8.4 Resultado

La aplicación de dicha estrategia de prueba, condujo al siguiente resultado:

En el nivel de pruebas de unidad se comprobó el correcto funcionamiento de los componentes de la solución de manera independiente.

Se comprobó la integración satisfactoria del módulo Eventos y titulaciones con los módulos: Tesis y Títulos, Estructura y composición, Seguridad y Carrera. Todo ello a través de la aplicación de los diseños de casos de prueba propuestos.

En el proceso de evaluación del nivel de pruebas del sistema, se aplicaron tres iteraciones de pruebas. De un total de 59 funcionalidades validadas, en la primera iteración se obtuvieron nueve no conformidades, que se corrigieron; en la segunda iteración cinco no conformidades, a las cuales se les dio solución y en la tercera y última iteración se detectó una no conformidad, que de igual manera fue resuelta. En la gráfica se muestra la estadística en cuanto a no conformidades por funcionalidades, en las distintas iteraciones.



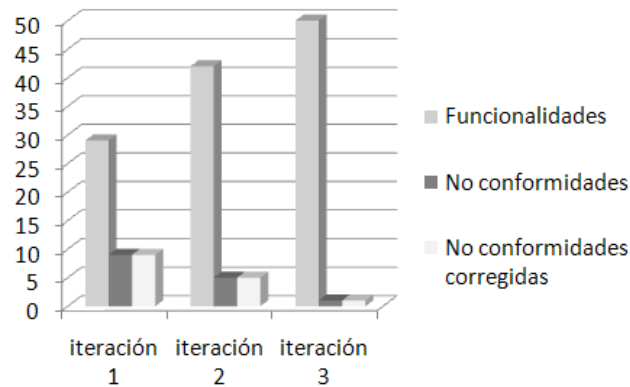


Figura 17 Resultado de pruebas en 3 iteraciones

Del total de 15 no conformidades identificadas en las tres iteraciones, el 66,6% son de validación, el 34,4% de interfaz. **Ver Figura 23.**



Figura 18 Porcentaje de no conformidades

3.9 Conclusiones

A partir de la especificación de los requerimientos y el diseño de la propuesta de solución se procedió a las dos últimas etapas del desarrollo del trabajo. En este capítulo se identificaron las técnicas, estándares y estilos para concretar la implementación. La solución fue desarrollada en el lenguaje propuesto definiendo además su integración con el Sistema de Gestión Universitaria. Además, se mostraron interfaces del resultado final. Luego de construida la aplicación se realizaron pruebas unitarias, de sistema e integración utilizando los diseños de casos de prueba confeccionados, para la obtención de un resultado satisfactorio.



Conclusiones generales

En la presente investigación se identificó la necesidad de la creación de una solución informática para la gestión de la información de eventos en la Universidad de las Ciencias Informáticas. La definición de conceptos relacionados con el objeto de estudio y el análisis de sistemas nacionales y extranjeros, que informatizan la gestión de información en eventos universitarios, proporcionaron un basamento teórico a partir del cual se pudo diseñar una propuesta de solución.

Mediante la aplicación de las técnicas de ingeniería de requisitos seleccionadas se obtuvieron las funcionalidades que posibilitaron la construcción de una solución para la gestión de información en eventos como un proceso integrado sin obviar las reglas del negocio para cada tipo de evento.

La utilización de la arquitectura de software definida por el CENIA permitió desarrollar una solución robusta, genérica, configurable, segura e integrada completamente al SGU. Con esto se logró la centralización de la información referente a los eventos que se realizan en la UCI lo cual facilitará la obtención de reportes y estadísticas confiables para el apoyo a la toma de decisiones.



Recomendaciones

- Realizar pruebas de Stress y Carga para asegurar el correcto funcionamiento de la aplicación en condiciones extremas, la cantidad de datos que puede gestionar, definiendo de esta manera los límites de la misma.
- Implementar funcionalidades que permitan la integración con le Archivo Universitarioa, para facilitar un correcto control de versiones de los documentos que se generan en la gestión de eventos.
- Informatizar el proceso de gestión de eventos universitarios a partir de la solución desarrollada utilizando flujos de trabajo.



Referencias Bibliográficas

1. **Real Academia Española.** *Diccionario de la Lengua Española.* 22ª. Madrid : ESPASA-CALPE, 2001. pág. 2448. ISBN: 9788423968145.
2. *Sistema de Gestión de la Investigación en la Universidad de Talca, Chile.* **Palomo, Ivan F., Veloso, Carlos G. y Schmal, Rodolfo F.** 1, Talca : s.n., 2007, Información Tecnológica, Vol. 18, págs. 97-106.
3. *La edición electrónica y la gestión de trabajos académicos con Open Conference Systems (OCS).* **Maquilón Sánchez, Javier, Lillo Hidalgo, Vicente y Mirete Ruiz, Ana Belén.** 1, 2011, Revista Electrónica Interuniversitaria de Formación del Profesorado, Vol. 14, págs. 325-344. ISSN 1575-0965.
4. **Heredia Rodríguez, Reynaldo y Salcerio Paz, Emilio.** *Aplicación Web para el control del Proceso de Tesis de Grado en la Facultad 8.* Ciudad de la Habana : s.n., 2008.
5. *Congreso virtual de informática en salud: experiencias con la herramienta Plone.* **Díaz Martínez, Ana Gloria, y otros.** 6, Ciudad de La Habana : Editorial de Ciencias Médicas, diciembre de 2009, ACIMED, Vol. 20. ISSN 1561-2880.
6. **Mcdonald Landazuri, Bárbara. A.** *Definición de Perfiles en Herramientas de Gestión de Requisitos.* Madrid : Facultad de Informática Universidad Politécnica de Madrid, 2005.
7. **Escalona, M. y Koch, N.** *Ingeniería de Requisitos en Aplicaciones para la Web: Un estudio comparativo.* . Sevilla : Escuela Técnica Superior de Ingeniería Informática. Universidad de Sevilla., 2002.
8. **Universidad Politécnica de Madrid (UPM).** Unidad Docente de Ingeniería del Software (UDIS) . [En línea] 2004. http://is.ls.fi.upm.es/docencia/masterTI/ARS/docs/Manual_M2C1U11.pdf.
9. **Pressman, Roger S.** *Ingeniería de Software un Enfoque Práctico.* 7ma. Edición. New York : McGraw-Hill Companies, Inc, 2010. ISBN/978-0-07-337 597-7.
10. **Espinosa Reinoso, Jorge Arturo.** *Análisis de los modelos de calidad de software.* Guayaquil : s.n., 2007.
11. **Chrissis, Mary Beth, Konrad, Mike y Shrum, Sandy.** *Guía para la integración de procesos.* USA : PearsonEducación, S.A., 2009. ISBN/9788478290963.
12. **Beck, Kent.** Manifiesto Ágil. [En línea] 2001. http://es.wikipedia.org/wiki/Manifiesto_%C3%A1gil.
13. **Schach, Stephen R.** *Análisis y Diseño Orientado a Objetos con UML y el Proceso Unificado.* Cuauhtémoc : McGraw-Hill Interamericana Editores, S.A. de C.V. , 2004. ISBN: 970-10-4982-9 .
14. **Freund, Jakob, Rucker, Bernd y Hitpass, Bernhard.** *BPMN 2.0 Manual de Referencia y Guía Práctica.* Santiago de Chile : RPI Chile, 2011. ISBN: 978-956-345-182-5.



15. **Instituto Nacional de Estadística e Informática (INEI). Perú.** Herramientas CASE. El mejor soporte para el proceso de desarrollo de software. [En línea] 1999. www.inei.gob.pe/biblioineipub/bancopub/Inf/Lib5103/Libro.pdf.
16. **Netbeans.** Netbeans.org. [En línea] 2012. [Citado el: 15 de Marzo de 2012.] <http://netbeans.org/features/index.html>.
17. **Ramos María Jesús, Ramos Alicia, Montero Fernando.** *Sistemas gestores de bases de datos.(Ciclo formativo. Grado superior. Informática).* Madrid: S.A. MCGRAW-HILL / INTERAMERICANA, 2006. 9788448148799.
18. **PostgreSQL.** *PostgreSQL 8.4.1 Documentation.* Berkeley, California : PostgreSQL Global Development Group, 2009. págs. págs. 368-378, 2031-2033.
19. **CodeIgniter.** ELLISLAB INC. [En línea] 2011. [Citado el: 22 de Noviembre de 2011.] <http://www.codeigniter.com>.
20. **JQUERY.** THE JQUERY FOUNDATION. [En línea] 2011 . [Citado el: 21 de Noviembre de 2011.] <http://jquery.com>.
21. **Apache.** El servidor de web Apache: Introducción práctica: Apache 1.x y 2.0 alpha. [En línea] [Citado el: 6 de Diciembre de 2011.] <http://acsblog.es/articulos/trunk/LinuxActual/Apache/html/x31.html>.
22. **Ubcuos.com.** Ubcuos.com. [En línea] 8 de Mayo de 2010. [Citado el: 11 de Marzo de 2012.] <http://www.ubicuos.com/2010/05/08/como-instalar-pgadmin-3/>.
23. **JavaScript.** Concepto de javascript. Mozilla developer network. [En línea] [Citado el: 12 de Enero de 2012.] https://developer.mozilla.org/es/Gu%C3%ADa_JavaScript_1.5/Concepto_de_JavaScript.
24. **CSS.** Guía Breve de CSS. Sitio de la Oficina del W3C . [En línea] [Citado el: 7 de Diciembre de 2011.] <http://w3c.es/Divulgacion/GuiasBreves/HojasEstilo..>
25. **My php.net.** My php.net. [En línea] [Citado el: 12 de Diciembre de 2011.] <http://www.php.net/manual/es/intro-whatcando.php>.
26. **García Molina, Jesús, y otros.** De los Procesos del Negocio a los Casos de Uso . [En línea] [Citado el: 3 de Febrero de 2012.] <http://www.cyta.com.ar/ta0604/v6n4a1.htm>.
27. **Bass, Clements y Kazman.** <http://www.mitecnologico.com/Main/DescomposicionModular>. [En línea] 1998. [Citado el: 27 de Febrero de 2012.] <http://www.mitecnologico.com/Main/DescomposicionModular>.
28. **Webcom.mx.** Consultoría Web, Arquitectura de la información . [En línea] © Web Comunicaciones S.C. [Citado el: 27 de Febrero de 2012.] <http://www.webcom.mx/spip.php?article102>.



29. **Mejía Alvarez, Dr. Pedro, Cova Suazo, Nancy Noemí y Pérez Reséndiz, Marisol.** Arquitectura de Software. [En línea] [Citado el: 28 de Febrero de 2012.] delta.cs.cinvestav.mx/~pmejia/softeng/Arquitecturas.ppt.
30. **Ivanex.** Ivanex. Sistema de selección de las rutas optimas. Patron de Arquitectura y Framework. [En línea] 28 de Abril de 2008. [Citado el: 28 de Febrero de 2012.] <http://ivanex.wikidot.com/patron-arquitectura>.
31. **Bedoya, Andrés.** Geek The Planet. *Patrones de Diseño, ¿qué son y para qué sirven?* [En línea] [Citado el: 9 de Marzo de 2012.] <http://geektheplanet.net/5461/patrones-de-diseno-%C2%BFque-son-y-para-que-sirven.xhtml>.
32. **García Chávez, Carlos Alberto.** mailxmail.com. [En línea] Abril de 2005. [Citado el: 1 de Marzo de 2012.] <http://www.mailxmail.com/curso-diseno-base-datos-relacionales/sistemas-bases-datos..>
33. **Garlan, David y Shaw, Mary.** An Introduction to Software Architecture. [En línea] 1994. [Citado el: 3 de Marzo de 2012.] www.cs.cmu.edu/afs/cs/.../intro_softarch.pdf. CMU-CS-94-166.
34. **Acosta, Lindsay.** Programacion, UNIDAD III: TECNICAS DE PROGRAMACION. [En línea] miércoles 27 de abril de 2011. [Citado el: 4 de Marzo de 2012.] <http://programacioniagmb.blogspot.com/2011/04/unidad-iii-tecnicas-de-programacion.html>.
35. **Lenguajes de Programación.** Lenguajes de Programación. [En línea] 2009. [Citado el: 4 de marzo de 2012.] <http://www.lenguajes-de-programacion.com/programacion-orientada-a-objetos.shtml>.
36. **Natalia Juristo, Ana M. y Sira Vegas, Moreno.** *TÉCNICAS DE EVALUACIÓN DE SOFTWARE.* Octubre, 2005.
37. **slideshare.** Pruebas de Software.Control de calidad del software. [En línea] 4 de Noviembre de 2009. [Citado el: 17 de Febrero de 2012.] <http://www.slideshare.net/cliceduca/pruebas-de-software-2420588>.
38. **Blaha, Michael.** *Patterns Of Data Modeling.* Filadelfia : Taylor & Francis, 2010. pág. Capítulo 1. ISBN: 9781439819890.
39. **Departamento de Lenguajes y Sistemas Informáticos.** Ingeniería de Requisitos en Aplicaciones para la Web –Un estudio comparativo. . [En línea]
40. **Pinto, P.** Autorneto. [En línea] 22 de Mayo de 2008. [Citado el: 4 de Marzo de 2012.] <http://autorneto.com/tecnologia/software/tecnicas-de-programacion>.



Bibliografía consultada

- Álvarez Sara.** *Arquitectura cliente-servidor.* <http://www.desarrolloweb.com/articulos/arquitectura-cliente-servidor.html>. 20 de Agosto del 2007.
- Bustamante Montes de Oca Sánchez de Lic. Antonio.** *Arquitectura de información y usabilidad: nociones básicas para los profesionales de la información.* http://bvs.sld.cu/revistas/aci/vol12_6_04/aci04604.htm#cargo. Noviembre 2004.
- Codina L.** *El Profesional de la Información.* <http://www.elprofesionaldelainformacion.com/>. Mayo 2003
- Hernández León, R. A., & Coello González, S.** *El paradigma cuantitativo de la investigación científica.* Ciudad de la Habana 2002.
- Labarrere Sarduy, Alberto F. y Vargas Alfaro Ana Tani.** *Tesis en opción al Grado Científico de Doctor en Ciencias de la Educación.* Pinar del Rio, Cuba 2002.
- Leyva Samada Lisandra Isabel.** Flujo de Investigación para la Metodología Ágil SX. Mayo 2009
- Pressman Roger.** Ingeniería en sistemas un enfoque práctico.
- Unicornios.** Guión de la Metodología Scrum & XP Unicornios Methodology SXP Methodology v 2.0



Glosario de términos

ASD: *Adaptive Software Development* es el modelo de implementación de patrones ágiles para desarrollo de software.

ISO-9000: designa un conjunto de normas sobre calidad y gestión continua de calidad.

CMM: el Modelo de Madurez de Capacidades o CMM (*Capability Maturity Model*), es un modelo de evaluación de los procesos de una organización.

C++: es un lenguaje de programación.

XML: *eXtensible Markup Language*, es un metalenguaje extensible de etiquetas.

Python: es un lenguaje de programación de alto nivel.

Ruby: es un lenguaje de programación interpretado, reflexivo y orientado a objetos.

CASE: *Computer Aided Software Engineering*, son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software.

API: *Application Programming Interface*, es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Mac OS: *Macintosh Operating System*, es el nombre del sistema operativo creado por Apple para su línea de computadoras Macintosh.

Solaris: es un sistema operativo de tipo Unix.

HTML: *Hyper Text Markup Language*, es el lenguaje de marcado predominante para la elaboración de páginas web.

Indentación: es un anglicismo (de la palabra inglesa *indentation*) de uso común en informática que significa mover un bloque de texto hacia la derecha insertando espacios o tabuladores para separarlo del texto adyacente.

Swing: el paquete Swing es parte de la *Java Foundation Classes*, en la plataforma Java que provee facilidades para ayudar a la gente a construir interfaces de usuario.

SQL Server: es un sistema para la gestión de bases de datos.

Licencia BSD: es la licencia original de una distribución de Software: *Berkeley Software Distribution*, que acabó convirtiéndose en un derivativo de UNIX

Rollbacks: en tecnologías de base de datos, es una operación que devuelve a la base de datos a algún estado previo.

Plone: es un potente y flexible Sistema de Gestión de Contenidos (CMS) que es fácil de instalar, utilizar y ampliar.

Zope: es una herramienta que sirve para construir sitios web.



Dublin Core: es un modelo de metadatos elaborado y auspiciado por la DCMI (Dublin Core Metadata Initiative).

Workflow: es el estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas.

Windows: es un sistema operativo con interfaz gráfica para computadoras personales propiedad de la empresa Microsoft. Windows es el sistema operativo más utilizado en el mundo.

Linux: es un Unix libre, es decir, un sistema operativo, como el Windows o el MS-DOS.

BSD: es un sistema operativo derivado del sistema Unix nacido a partir de los aportes realizados a ese sistema por la Universidad de California en Berkeley.

Popud: un pop-up o ventana pop-up o ventana emergente, es una ventana nueva que aparece de repente en la pantalla de tu ordenador.

jQuery UI: es una biblioteca de componentes para el framework jQuery que le añaden un conjunto de plug-ins, widgets y efectos visuales para la creación de aplicaciones web.

Tooltip: es una herramienta de ayuda visual, que funciona al situar el cursor sobre algún elemento gráfico, mostrando una ayuda adicional para informar al usuario de la finalidad del elemento sobre el que se encuentra.

Mainframes: computadora grande, poderosa y costosa utilizada principalmente en empresas que necesitan procesar gran cantidad de datos o soportar gran cantidad de usuarios.

Script CGI: denominado pasarelas o CGI scripts consisten generalmente de una serie de instrucciones escritas en un lenguaje de programación como C o PERL que procesan la petición de un navegador.

FastCGI: es un protocolo para interconectar programas interactivos con un servidor web.

Lighttpd: (pronunciado lighty) es un servidor web diseñado para ser rápido, seguro, flexible, y fiel a los estándares.

Nginx: (pronunciado en inglés “engine X”) es un servidor web/proxy inverso ligero de alto rendimiento y un proxy para protocolos de correo electrónico (IMAP/POP3).

DOM: es la estructura de objetos que genera el navegador cuando se carga un documento y se puede alterar mediante Javascript para cambiar dinámicamente los contenidos y aspecto de la página.

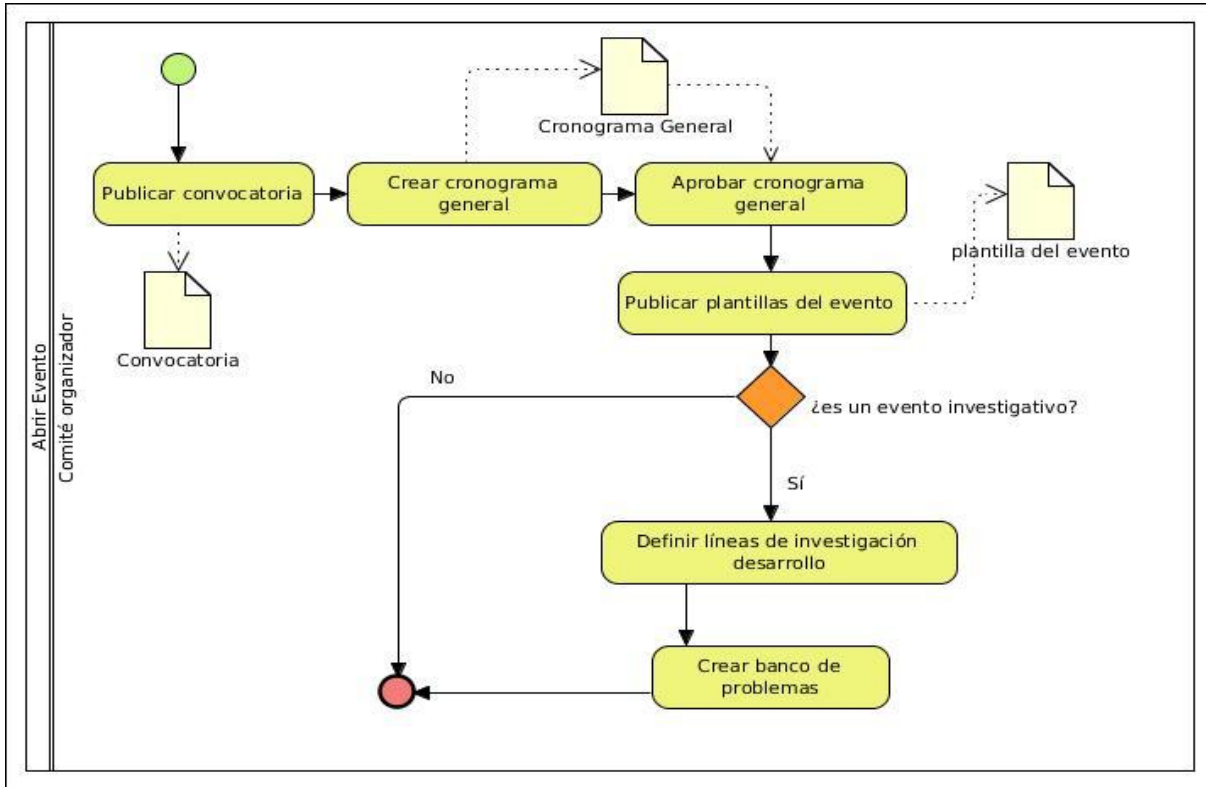
Ajax: acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications).

CMS: Sistema Gestor de Contenidos es un sistema que nos permite gestionar contenidos. En líneas generales, permitiría administrar contenidos en un medio digital y para el caso particular que nos ocupa, permitiría gestionar los contenidos de una web.

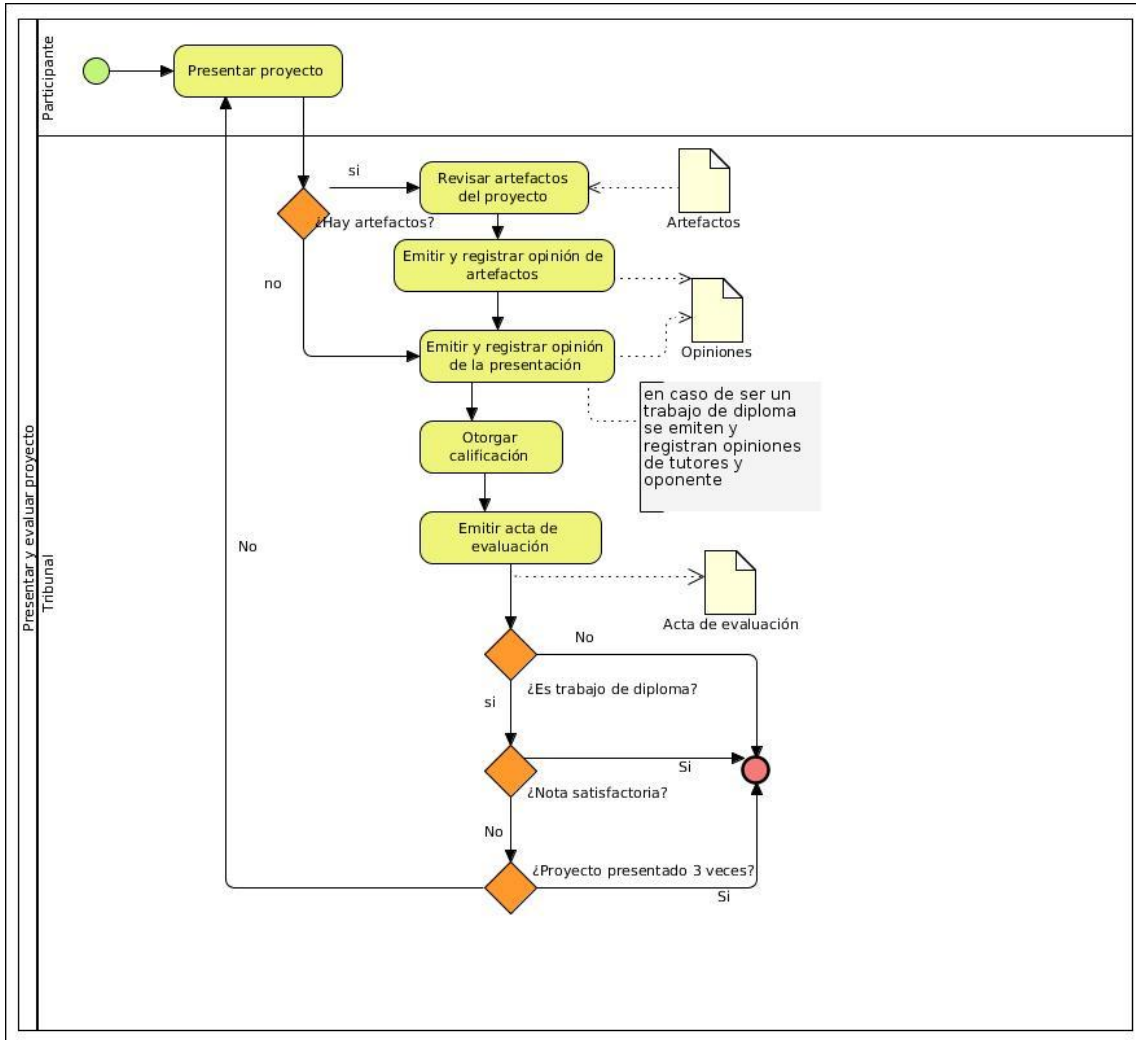


Anexos

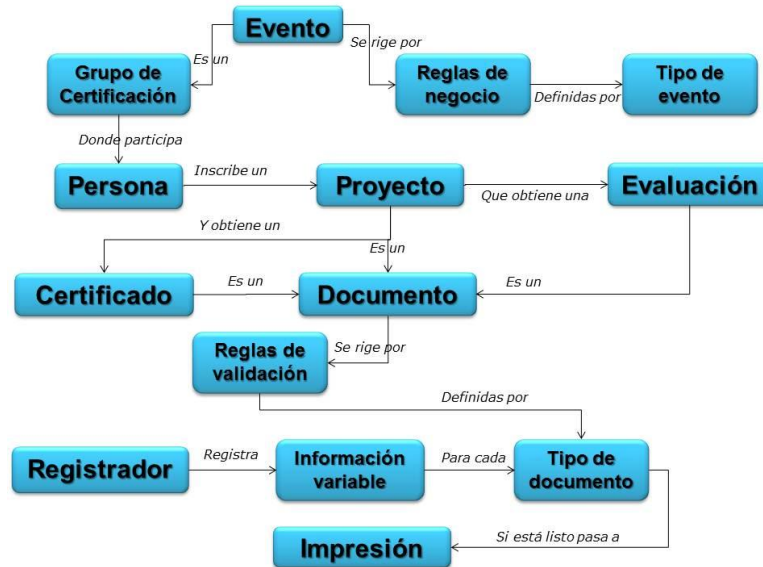
Anexo 1. Modelado de Procesos del Negocio con BPMN. Subproceso: Abrir evento.



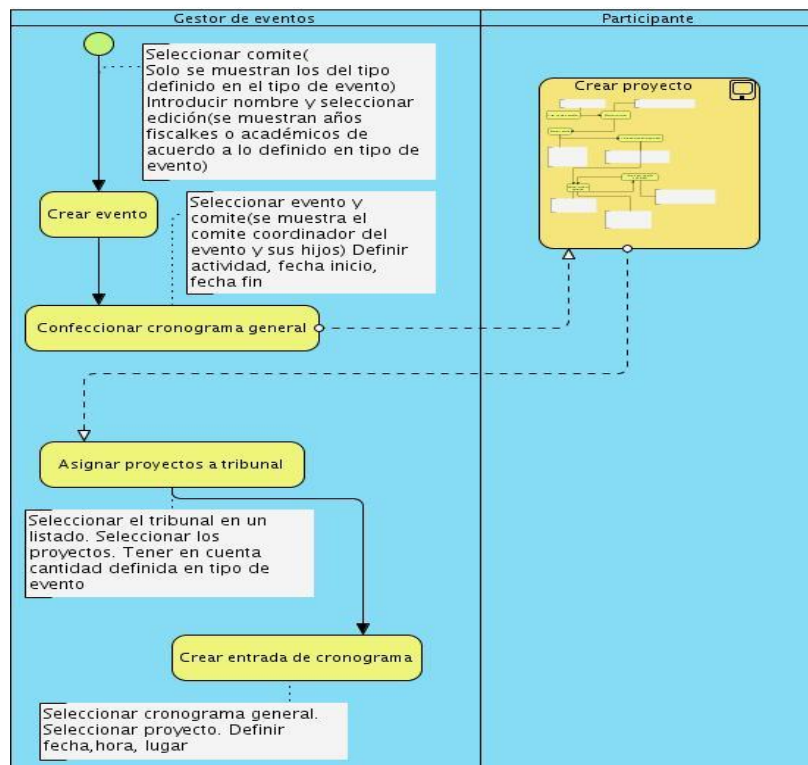
Anexo 2. Modelado de Procesos del Negocio con BPMN. Subproceso: Presentar y evaluar proyecto.



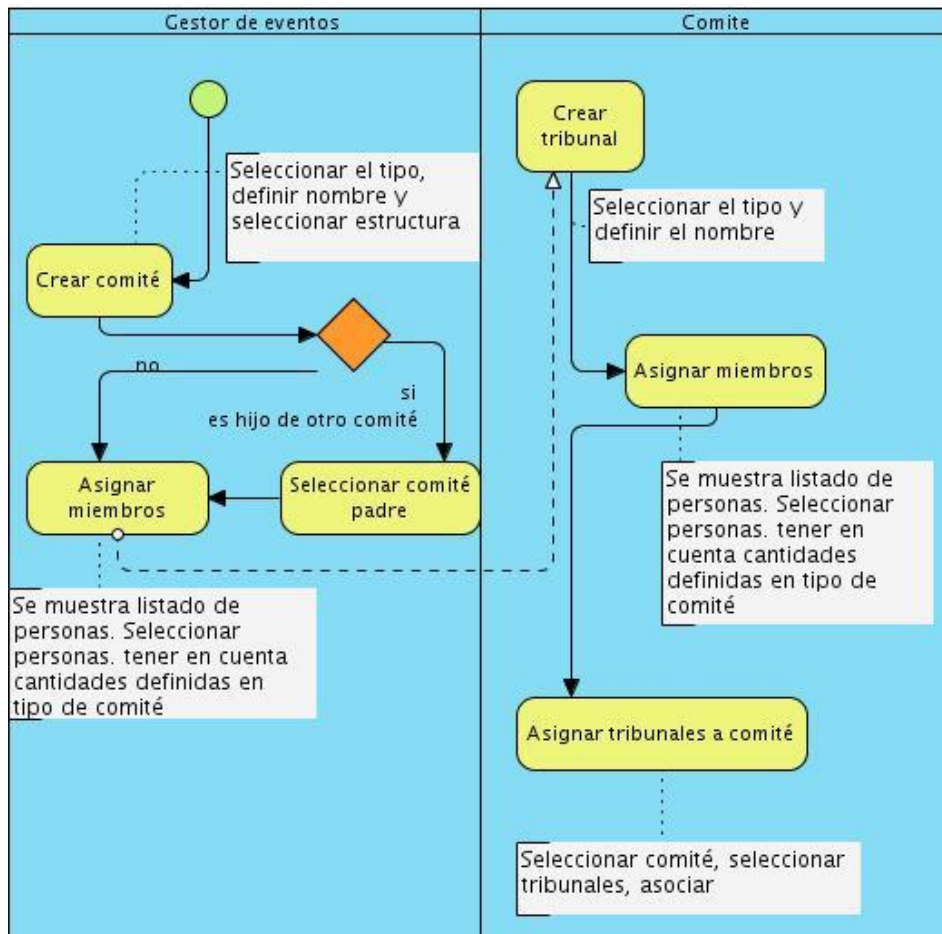
Anexo 3. Conceptualización inicial.



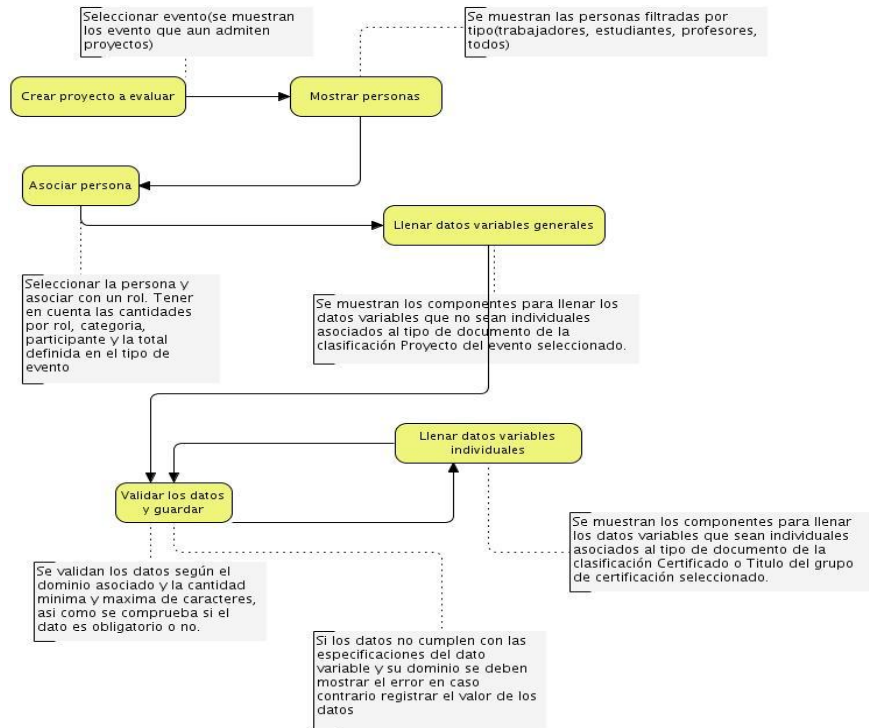
Anexo 4. Diagrama que indica cómo se gestiona un evento en el sistema.



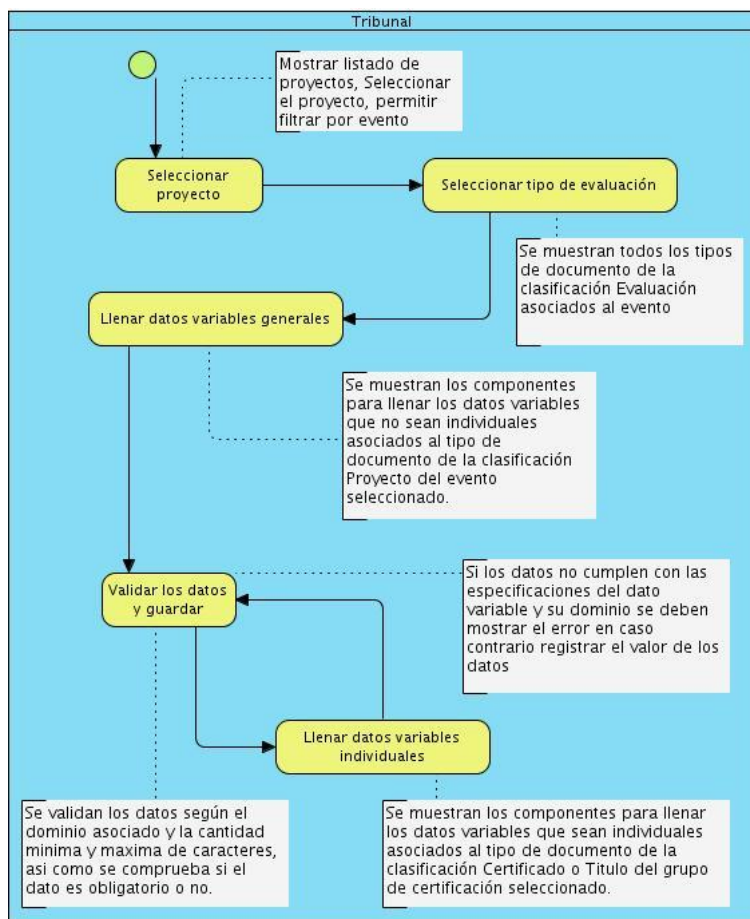
Anexo 5. Diagrama que indica cómo se gestiona un tribunal y un comité en el sistema.



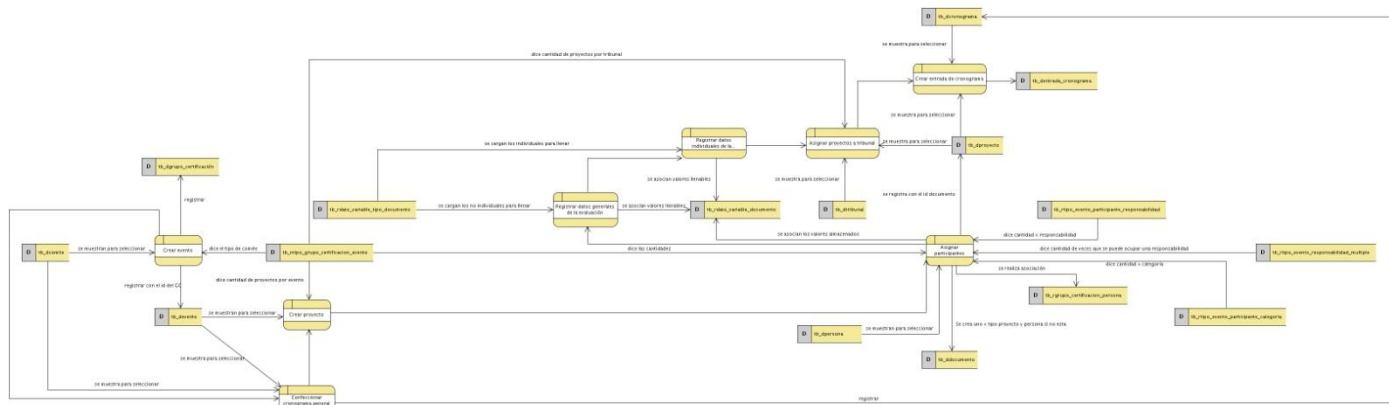
Anexo 6. Diagrama que indica cómo se registra un proyecto en el sistema.



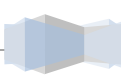
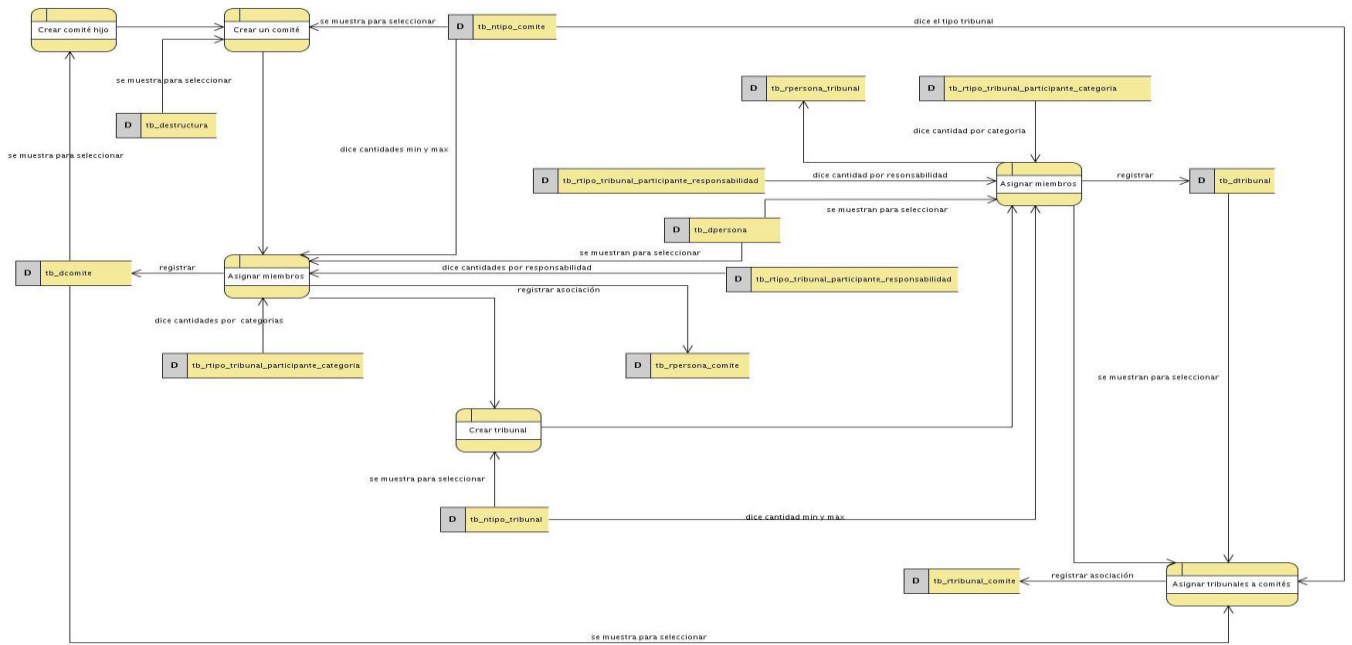
Anexo 7. Diagrama que indica cómo se registra una evaluación para un proyecto en el sistema.



Anexo 8. Diagrama flujo de datos en la funcionalidad de gestionar un evento.



Anexo 9. Diagrama flujo de datos en la funcionalidad de gestionar tribunales y comités.



Anexo 10. Lista de verificación.

Requisitos Validados										Resultado
Requisitos Funcionales(RF)	¿El RF está establecido de manera clara?	¿La fuente del RF está identificada?	¿El enunciado final del RF ha sido examinado por la fuente original?	¿El RF está restringido en términos cuantitativos?	¿El REQ viola alguna restricción del dominio del sistema?	¿El RF se puede probar?	¿El RF es rastreadable para los objetivos generales del sistema o producto?	¿La especificación está estructurada de una forma que conduzca a su comprensión, referencia y traducción fácil en productos de trabajo más técnicos?	¿Cuáles otros RF están relacionados con este?	Decisión
RFET_1 Crear evento	SI	SI	SI	NO	NO	SI	SI	SI	RFET_2, RFET_3, RFET_4, RFET_5, RFET_6, RFET_14, RFET_15, RFET_16	Aceptado
RFET_2 Modificar evento	SI	SI	SI	NO	NO	SI	SI	SI	RFET_3, RFET_4, RFET_5, RFET_6, RFET_15, RFET_16	Aceptado
RFET_3 Mostrar eventos	SI	SI	SI	NO	NO	SI	SI	SI	RFET_2, RFET_4	Aceptado
RFET_4 Ver detalles	SI	SI	SI	NO	NO	SI	SI	SI		Aceptado
RFET_5 Crear proyecto a presentar	SI	SI	SI	NO	NO	SI	SI	SI	RFET_6, RFET_7, RFET_8, RFET_9, RFET_14, RFET_33, RFET_34	Aceptado
RFET_6 Modificar	SI	SI	SI	NO	NO	SI	SI	SI	RFET_1, RFET_2, RFET_5	Aceptado
RFET_7 Mostrar proyectos a presentar	SI	SI	SI	NO	NO	SI	SI	SI	RFET_5, RFET_6, RFET_8	Aceptado
RFET_8 Ver detalles de proyecto a presentar	SI	SI	SI	NO	NO	SI	SI	SI	RFET_5, RFET_6, RFET_7	Aceptado
RFET_9 Asignar participantes a proyecto	SI	SI	SI	NO	NO	NO	SI	SI	RFET_5, RFET_6, RFET_7	Rechazado
RFET_10 Crear tribunal	SI	SI	SI	NO	NO	SI	SI	SI	RFET_11, RFET_12, RFET_13, RFET_14, RFET_19	Aceptado
RFET_11 Modificar tribunal	SI	SI	SI	NO	NO	SI	SI	SI	RFET_12, RFET_13, RFET_14	Aceptado
RFET_12 Mostrar tribunales	SI	SI	SI	NO	NO	SI	SI	SI	RFET_14, RFET_12, RFET_13	Aceptado
RFET_13 Ver detalles de tribunal	SI	SI	SI	NO	NO	SI	SI	SI		Aceptado
RFET_14 Asignar proyectos a tribunal	SI	SI	SI	NO	NO	NO	SI	SI	RFET_13	Rechazado
RFET_15 Crear comité	SI	SI	SI	NO	NO	NO	NO	SI	RFET_11, RFET_12, RFET_19, RFET_17, RFET_16, RFET_18, RFET_29, RFET_30	Rechazado
RFET_16 Modificar comité	SI	SI	SI	NO	NO	NO	NO	SI	RFET_11, RFET_12, RFET_19, RFET_17, RFET_18, RFET_29, RFET_30	Rechazado
RFET_17 Mostrar comités	SI	SI	SI	NO	NO	SI	SI	NO	RFET_19, RFET_18, RFET_16	Rechazado
RFET_18 Ver detalles de comité	SI	SI	SI	NO	NO	SI	SI	SI		Rechazado
RFET_19 Asignar tribunales a comité	SI	SI	SI	NO	NO	SI	SI	SI		Rechazado
RFET_20 Registrar evaluación de proyecto	SI	SI	SI	NO	NO	SI	SI	NO	RFET_36	Rechazado
RFET_21 Crear problema	SI	SI	SI	NO	NO	SI	SI	SI	RFET_22, RFET_23, RFET_24, RFET_25, RFET_26, RFET_27	Aceptado



RFET_22 Modificar problema	SI	SI	SI	NO	NO	SI	SI	SI	RFET_23, RFET_24, RFET_25, RFET_26	Aceptado
RFET_23 Mostrar problemas	SI	SI	SI	NO	NO	SI	SI	SI	RFET_22, RFET_25	Aceptado
RFET_24 Ver detalles de problema	SI	SI	SI	NO	NO	SI	SI	SI		Aceptado
RFET_25 Crear tema de investigación	SI	SI	SI	NO	NO	SI	SI	SI	RFET_38, RFET_39	Aceptado
RFET_26 Modificar tema de investigación	SI	SI	SI	NO	NO	SI	SI	SI	RFET_38, RFET_39	Aceptado
RFET_27 Mostrar temas de investigación	SI	SI	SI	NO	NO	SI	SI	SI		Aceptado
RFET_28 Ver detalles de tema de investigación	SI	SI	SI	NO	NO	SI	SI	SI		Aceptado
RFET_29 Crear cronograma general	SI	SI	SI	NO	NO	NO	SI	NO	RFET_30, RFET_31, RFET_32, RFET_33, RFET_34, RFET_35	Rechazado
RFET_30 Modificar cronograma general	SI	SI	SI	SI	NO	NO	SI	NO	RFET_31, RFET_32, RFET_33, RFET_34, RFET_35	Rechazado
RFET_31 Mostrar cronogramas	SI	SI	SI	SI	NO	NO	SI	NO	RFET_33, RFET_34, RFET_35, RFET_36	Rechazado
RFET_32 Ver detalles cronograma general	SI	SI	SI	SI	NO	SI	SI	NO		Rechazado
RFET_33 Crear entrada de cronograma	SI	SI	SI	SI	NO	SI	SI	NO		Rechazado
RFET_34 Modificar entrada de cronograma	SI	SI	SI	SI	NO	SI	SI	NO		Rechazado
RFET_35 Ver detalles entrada de cronograma	SI	SI	SI	SI	NO	SI	SI	NO		Rechazado
RFET_35 Mostrar registro de evaluaciones	SI	SI	SI	SI	NO	SI	SI	SI		Aceptado
RFET_37 Mostrar informes de temas de investigación	SI	SI	SI	SI	NO	SI	SI	SI		Aceptado
RFET_38 Crear informe de tema de investigación	SI	SI	SI	SI	NO	SI	SI	SI		Aceptado
RFET_39 Revisar	SI	SI	SI	SI	NO	SI	SI	SI		Aceptado
RFET_40 Crear tipo de evento	SI	SI	SI	SI	NO	SI	SI	SI	RFET_1, RFET_2, RFET_3, RFET_4, RFET_29, RFET_30, RFET_32, RFET_37, RFET_116	Aceptado
RFET_41 Modificar tipo de evento	SI	SI	SI	SI	NO	SI	SI	SI	RFET_32, RFET_33, RFET_34, RFET_35, RFET_73, RFET_74, RFET_36, RFET_6, RFET_42	Aceptado
RFET_42 Mostrar tipos de evento	SI	SI	SI	SI	NO	SI	SI	SI	RFET_41, RFET_43	Aceptado
RFET_43 Ver detalles de tipo de evento	SI	SI	SI	SI	NO	SI	SI	SI		Aceptado
RFET_44 Crear tipo de tribunal	SI	SI	SI	SI	NO	SI	SI	SI	RFET_19, RFET_10, RFET_11, RFET_12, RFET_13, RFET_48, RFET_49, RFET_45	Aceptado



RFET_45 Modificar tipo de tribunal	SI	SI	SI	NO	NO	SI	SI	SI	RFET_19,RFET_10,RFET_11,RFET_12,RFET_13,RFET_48,RFET_49	Aceptado
RFET_46 Mostrar tipos de tribunal	SI	SI	SI	NO	NO	SI	SI	SI	RFET_45,RFET_47	Aceptado
RFET_47 Ver detalles de tipo de tribunal	SI	SI	SI	NO	NO	SI	SI	SI		Aceptado
RFET_48 Crear tipo de comité	SI	SI	SI	NO	NO	SI	SI	SI	RFET_15,RFET_16,RFET_18,RFET_4	Aceptado
RFET_49 Modificar tipo de comité	SI	SI	SI	NO	NO	SI	SI	SI	RFET_15,RFET_16,RFET_18	Aceptado
RFET_50 Mostrar tipos de comité	SI	SI	SI	SI	NO	SI	SI	SI	RFET_51,RFET_49	Aceptado
RFET_51 Ver detalles de tipo de comité	SI	SI	SI	SI	NO	SI	SI	SI		Aceptado
RFET_52 Crear línea de investigación-desarrollo	SI	SI	SI	SI	NO	SI	SI	SI	RFET_21,RFET_22,RFET_24,RFET_25,RFET_26,RFET_28,RFET_53	Aceptado
RFET_53 Modificar línea de investigación-desarrollo	SI	SI	SI	SI	NO	SI	SI	SI	RFET_21,RFET_22,RFET_24,RFET_25,RFET_26,RFET_28	Aceptado
RFET_54 Mostrar línea	SI	SI	SI	SI	NO	SI	SI	SI	RFET_53,RFET_55	Aceptado
RFET_55 Ver detalles de línea de investigación-desarrollo	SI	SI	SI	NO	NO	SI	SI	SI		Aceptado
RFET_56 Crear responsabilidad	SI	SI	SI	NO	NO	SI	SI	SI	RFET_40,RFET_41,RFET_42,RFET_44,RFET_46,RFET_47,RFET_49,RFET_115,RFET_116,RFET_118,RFET_119,RFET_120,RFET_122,RFET_123,RFET_124,RFET_126,RFET_133	Aceptado
RFET_57 Modificar responsabilidad	SI	SI	SI	NO	NO	SI	SI	SI	RFET_9,RFET_10,RFET_11,RFET_13,RFET_15,RFET_16,RFET_18,RFET_40,RFET_41,RFET_43,RFET_44,RFET_45,RFET_47,RFET_48,RFET_124,RFET_51	Aceptado
RFET_58 Mostrar responsabilidad	SI	SI	SI	NO	NO	SI	SI	SI	RFET_57,RFET_59	Aceptado
RFET_59 Ver detalles responsabilidad	SI	SI	SI	NO	NO	SI	SI	SI		Aceptado

Anexo 10.1 Lista de verificación segunda iteración



Requisitos Validados										Resultado
Requisitos Funcionales(RF)	¿El RF está establecido de manera clara?	¿La fuente del RF está identificada?	¿El enunciado final del RF ha sido examinado por la fuente original?	¿El RF está restringido en términos cuantitativos?	¿El REQ viola alguna restricción del dominio del sistema?	¿El RF se puede probar?	¿El RF es rastreado para los objetivos generales del sistema o producto ?	¿La especificación está estructurada de una forma que conduzca a su comprensión, referencia y traducción fácil en productos de	¿Cuáles otros RF están relacionados con este?	Decisión
RFET_40 Asignar participantes a proyecto	SI	SI	SI	NO	NO	SI	SI	SI	RFET_5,RFET_6,RFET_7	Aceptado
RFET_45 Asignar proyectos a tribunal	SI	SI	SI	NO	NO	SI	SI	SI	RFET_13	Aceptado
RFET_46 Crear comité	SI	SI	SI	NO	NO	SI	NO	SI	RFET_11,RFET_12, RFET_19,RFET_17,RFET_16, RFET_18, RFET_29,RFET_30	Aceptado
RFET_47 Modificar comité	SI	SI	SI	NO	NO	SI	NO	SI	RFET_11,RFET_12, RFET_19,RFET_17,RFET_18, RFET_29,RFET_30	Aceptado
RFET_48 Mostrar comités	SI	SI	SI	NO	NO	SI	SI	SI	RFET_19,RFET_18,RFET_16	Aceptado
RFET_49 Ver detalles de comité	SI	SI	SI	NO	NO	SI	SI	SI		Aceptado
RFET_50 Asignar tribunales a comité	SI	SI	SI	NO	NO	SI	SI	SI		Aceptado
RFET_51 Registrar evaluación de	SI	SI	SI	NO	NO	SI	SI	SI	RFET_36	Aceptado
RFET_73 Crear cronograma general	SI	SI	SI	NO	NO	SI	SI	SI	RFET_30,RFET_31, RFET_32,RFET_33, RFET_34,RFET_35	Aceptado
RFET_74 Modificar cronograma general	SI	SI	SI	SI	NO	SI	SI	SI	RFET_31, RFET_32,RFET_33,RFET_34, RFET_35	Aceptado
RFET_75 Mostrar cronogramas	SI	SI	SI	SI	NO	SI	SI	SI	RFET_33,RFET_34, RFET_35,RFET_36	Aceptado
RFET_76 Ver detalles	SI	SI	SI	SI	NO	SI	SI	SI		Aceptado
RFET_77 Crear entrada de	SI	SI	SI	SI	NO	SI	SI	SI		Aceptado
RFET_78 Modificar entrada de	SI	SI	SI	SI	NO	SI	SI	SI		Aceptado
RFET_79 Ver detalles entrada de cronograma	SI	SI	SI	SI	NO	SI	SI	SI		Aceptado

Anexo 11. Mapa de navegación de la solución

+ Mediante esta vista se accede al mostrar, crear, modificar y ver detalles.

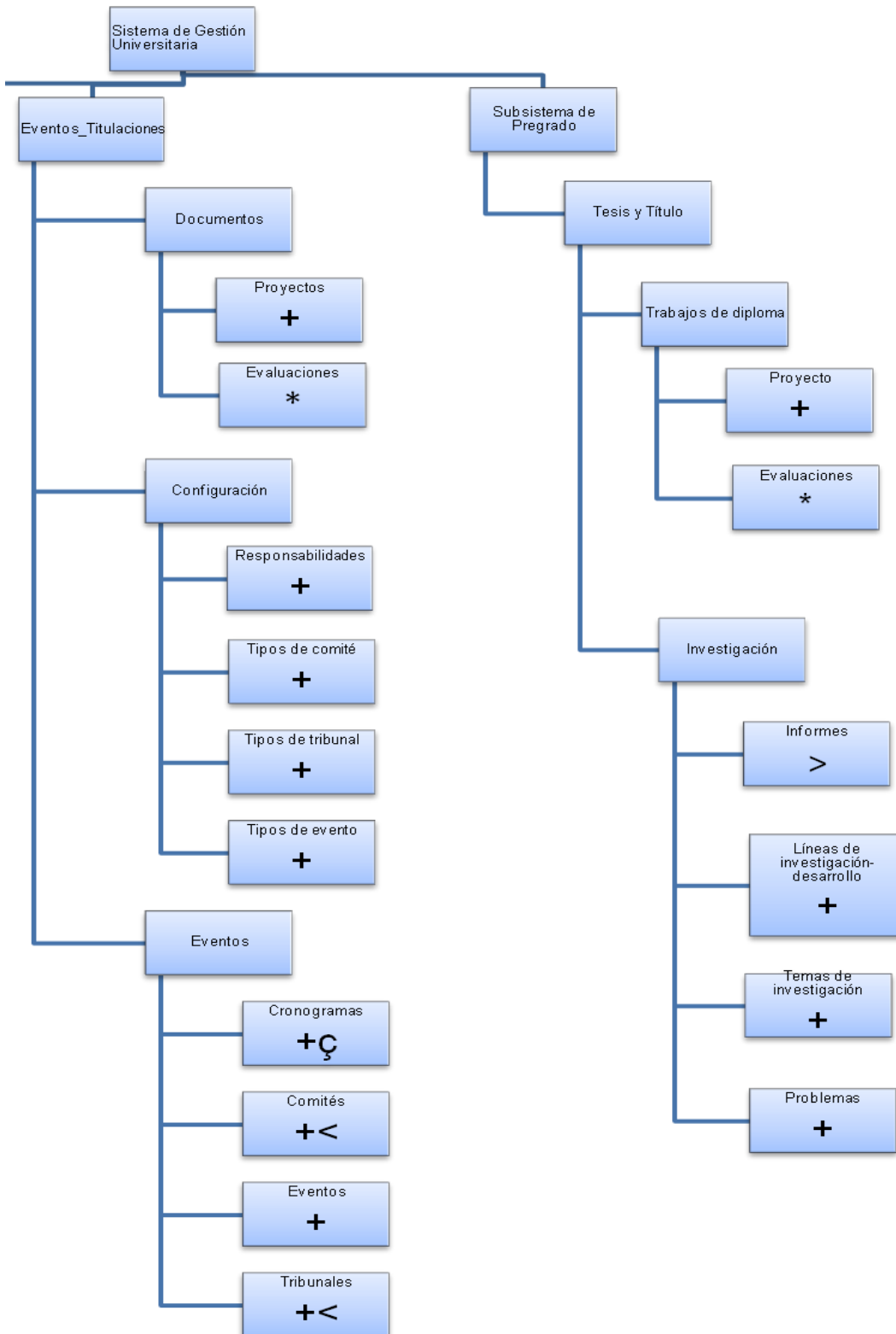
+< Además del gestionar tiene también un asignar.

> Presenta las vistas ver detalles, revisar y crear informe

*Contiene las interfaces registrar y mostrar registro.

+ç Además del gestionar tiene también gestionar entrada de cronograma y crear cronograma hijo.





Anexo 12. Modelo físico de la base de datos

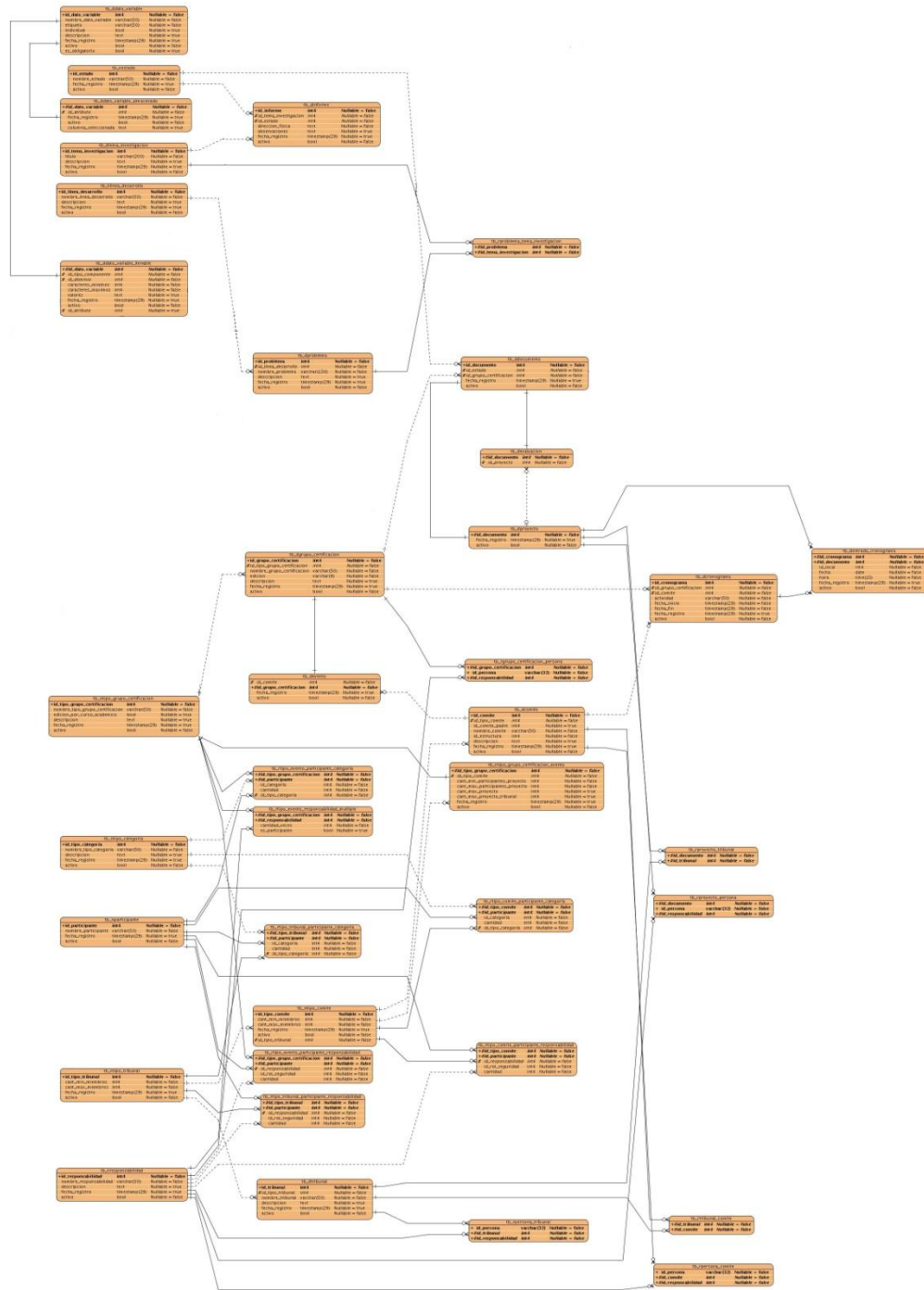


Figura 19 Modelo físico de base de dato



Anexo 13. Código modificar comité

```

public function modificar()
{
    $all_post = $this->input->all_post(true); (1)
    $id_comite = $all_post['id']; (1)
    if(empty($id_comite)) (1)
    {
        throw new Exception_Error('SYS007'); (2)
    }
    $comite = $this->comite_lib->obtenercomiteDadoIdcomite($id_comite); (3)
    if(empty($comite)) (3)
    {
        throw new Exception_Error('SYS006'); (4)
    }
    $this->template->set_data('id_comite', $comite->id_comite); (5)
    $this->template->set_data('nombre_comite', $comite->nombre_comite); (5)
    $this->template->set_data('descripcion', $comite->descripcion); (5)
    $tipos_comite = $this->comite_lib->obtenertipo_comite(); (5)
    $tipos_comite = convert_assoc_array($tipos_comite, 'id_tipo_comite', 'nombre_comite', true); (5)
    $this->template->set_data('tipo_comite', $tipos_comite); (5)
    $this->template->set_data('id_tipo_comite', $comite->id_tipo_comite); (5)
    //Filtros
    $area = $this->comite_lib->obtenerAreas(''); (5)
    $area = convert_assoc_array($area, 'id_estructura', 'nombre_estructura', true); (5)
    $facultades = $this->obnertFacultades(); (5)
    $facultad = convert_assoc_array($facultades, 'id_estructura', 'nombre_estructura', true); (5)
    $this->template->set_data('area', $area); (5)
    $this->template->set_data('facultad', $facultad); (5)
    if($comite->activo == 't') (5)
        $this->template->set_data('tipo_comite', TRUE); (6)
    else
        $this->template->set_data('tipo_comite', FALSE); (7)
    echo $this->template->render('comite/modificar_comite_view'); (8)
} (8)

```

Anexo 14. Código modificar tribunal.

```

public function modificartribunal() {
    if ($this->input->is_post_back(array(nombre_tribunal))) { (1)
        $post_vars = $this->input->all_post(true); (2)
        if ($this->tribunal_lib->modificartribunal($post_vars) != false) (2)
            $this->message('SYS002'); (4)
        else
            throw new Exception_Error('SYS006'); (5)
    }
    else
        throw new Exception_Error('SYS007'); (3)
} (6)

```



Anexo 15. Diseños de casos de prueba de integración.

Condiciones de entrada

Elemento		Estados		
Base de datos(BD)		Activa, con problemas.		
Tipos de documento		Ausente, presente, vacío.		
Tipos de evento		Con problemas, activo.		
Caso de Prueba 3				
Componente que se integra: Tipos de evento				
Componente a integrar: Tipos de documento				
Condiciones de ejecución: El módulo Eventos y titulaciones debe tener introducidos los datos en la base de datos y debe existir conexión con la misma.				
Objetivo de la prueba: Verificar que al crear y modificar un tipo de evento se pueda guardar como dato del mismo el tipo de documento.				
Escenarios	Entradas	Condiciones de entrada	Salidas	Descripción
Se muestran los tipos de documento correctamente.		BD activa, Tipos de documento presente, Tipos de evento activo.	El sistema muestra en un campo de selección los tipos de documento existentes.	El sistema obtiene los tipos de documento mostrándolos en campo de selección.
No se muestran los tipos de documento.		BD con problemas, Tipos de documento presente, Tipos de eventos activo.	El sistema muestra un mensaje de error.	El sistema no muestra los tipos de documento existentes.
		BD activa, Tipos de documento ausente, Tipos de evento activo.		
		BD activa, Tipos de documento presente, Tipos de evento con problemas.		
Se guardan el tipo de documento como información	Se selecciona el tipo de documento.	BD activa, Tipos de documento presente, Tipos de evento activo.	El sistema muestra un listado vacío.	El sistema al no obtener ningún dato muestra un listado vacío.
		BD activa, Tipos de documento seleccionado como dato de la tabla correspondiente al tipo	El sistema almacena el identificador del tipo de documento seleccionado como dato de la tabla correspondiente al tipo	El usuario selecciona el tipo de documento. El sistema



del tipo de evento.			de evento.	almacena el dato como información del tipo de evento en la tabla correspondiente de la base de datos.
No se guarda el tipo de documento seleccionado como dato del tipo de evento.	Se selecciona el tipo de documento.	BD con problemas, Tipos de documento presente, Tipos de evento activo. BD activa, Tipos de documento presente, Tipos de evento con problemas.	El sistema muestra un mensaje de error.	Una vez seleccionado o el tipo de documento, el usuario escoge la opción aceptar. El sistema no guarda el dato, muestra un mensaje de error.

Anexo 15.1. Diseños de casos de prueba de integración caso prueba 2

Condiciones de entrada

Elemento		Estados		
Base de datos(BD)		activa, con problemas.		
Datos variables		ausente, presente, vacío.		
Tipos de documento		con problemas, activo.		
Caso de Prueba 2				
Componente que se integra: Tipos de documento				
Componente a integrar: Datos variables				
Condiciones de ejecución: El módulo Eventos y titulaciones debe tener introducidos los datos en la base de datos y debe existir conexión con la misma.				
Objetivo de la prueba: Verificar que al crear y modificar un tipo de documento se puedan guardar como información del mismo los datos variables seleccionados.				
Escenarios	Entradas	Condiciones de entrada	Salidas	Descripción
Se muestran los datos variables correctamente.		BD activa, Datos variables presente, Tipos de documento activo.	El sistema muestra en un listado los datos variables existentes.	El sistema obtiene los datos variables



				mostrando los en un listado.
No se muestran los datos variables.		BD con problemas, Datos variables presente, Tipos de documento activo.	El sistema muestra un mensaje de error.	El sistema no muestra los datos variables existentes.
		BD activa, Datos variables ausente, Tipos de documento activo.		
		BD activa, Datos variables presente, Tipos de documento con problemas.		
		BD activa, Datos variables vacío, Tipos de documento activo.	El sistema muestra un listado vacío.	El sistema al no obtener ningún dato muestra un listado vacío.
Se guardan los datos variables como información del tipo de documento.	Se asocian los datos variables.	BD activa, Datos variables presente, Tipos de documento activo.	El sistema almacena los identificadores de los datos variables asociados como información de la tabla correspondiente al tipo de documento.	El usuario asocia los datos variables. El sistema almacena los datos variables en la base de datos como información correspondiente al tipo de documento.
No se guardan los datos variables asociados como información referente al tipo de documento.	Se asocian los datos variables.	BD con problemas, Datos variables presente, Tipos de documento activo	El sistema muestra un mensaje de error.	Una vez asociados los datos variables, el usuario escoge la opción aceptar. El sistema no guarda
		BD activa, Datos variables presente, Tipos de documento con problemas.		



				los datos, muestra un mensaje de error.
--	--	--	--	---

Anexo 15.2. Diseños de casos de prueba de integración caso prueba 3.

Condiciones de entrada

Elemento	Estados
Base de datos(BD)	Activa, con problemas.
Temas de investigación	Ausente, presente, vacío.
Datos variables	Con problemas, activo.
ioc	Con problemas, vacío, activo.

Caso de Prueba 1				
Componente que se integra: Datos variables				
Componente a integrar: Temas de investigación del módulo Tesis y Títulos				
Condiciones de ejecución: El módulo Tesis y Títulos debe tener introducidos los datos en la base de datos y debe existir conexión con la misma.				
Objetivo de la prueba: Verificar que se puedan almacenar los temas de investigación creados en el módulo Tesis y Títulos como dato variable en la tabla correspondiente.				
Escenarios	Entradas	Condiciones de entrada	Salidas	Descripción
Se muestran los temas de investigación correctamente.		BD activa, Temas de investigación presente, Datos variables activo, ioc activo.	El sistema muestra en un listado los temas de investigación existentes.	El sistema obtiene los temas de investigación mostrándolos en un listado.
No se muestran los temas de investigación.		BD con problemas, Temas de investigación presente, Datos variables activo, ioc activo.	El sistema muestra un mensaje de error.	El sistema no muestra los temas de investigación existentes.
		BD activa, Temas de investigación ausente, Datos variables activo, ioc activo.		
		BD activa, Temas de investigación presente, Datos variables con		



		problemas, ioc activo.		
		BD activa, Temas de investigación presente, Datos variables activo, ioc con problemas.		
		BD activa, Temas de investigación presente, Datos variables activo, ioc vacío.		
		BD activa, Temas de investigación vacío, Datos variables activo, ioc activo.	El sistema muestra un listado vacío.	El sistema al no obtener ningún dato muestra un listado vacío.
Se guardan los temas de investigación como dato variable.	Se asocian los temas de investigación.	BD activa, Temas de investigación presente, Datos variables activo, ioc activo.	El sistema almacena los identificadores de los temas de investigación asociados como información de la tabla correspondiente al dato variable.	El usuario asocia los temas de investigación. El sistema almacena los temas de investigación en la base de datos como información correspondiente al dato variable.
No se guardan los temas de investigación asociados como información referente al dato variable.	Se asocian los temas de investigación.	BD con problemas, Temas de investigación presente, Datos variables activo, ioc activo.	El sistema muestra un mensaje de error.	Una vez asociados los temas de investigación, el usuario escoge la
		BD activa, Temas de investigación		



		presente, Datos variables con problemas, ioc activo.		opción aceptar. El sistema no guarda los temas de investigac ión, muestra un mensaje de error.
		BD activa, Temas de investigación presente, Datos variables activo, ioc con problemas.		

Anexo 15.3. Diseños de casos de prueba de integración caso prueba 4.

Condiciones de entrada

Elemento	Estados
Base de datos(BD)	Activa, con problemas.
Curso académico	Ausente, presente, vacío.
Eventos	Con problemas, activo.
ioc	Con problemas, vacío, activo.

Caso de Prueba				
Componente que se integra: Eventos del módulo Eventos y titulaciones.				
Componente a integrar: Curso académico del módulo Carrera				
Condiciones de ejecución: El módulo Carrera debe tener introducidos los datos en la base de datos y debe existir conexión con la misma.				
Objetivo de la prueba: Verificar que los Cursos académicos del módulo Carrera se almacenen como información de Eventos en la base de datos del módulo Eventos y titulaciones.				
Escenarios	Entradas	Condiciones de entrada	Salidas	Descripción
Se muestran los cursos académicos correctamente.		BD activa, Curso académico presente, Eventos activo, ioc activo.	El sistema muestra en un campo de selección los cursos académicos existentes.	El sistema obtiene los cursos académicos del módulo Carrera mostrándolos en un campo de selección.
No se muestran los Cursos académicos.		BD con problemas, Curso académico	El sistema muestra un mensaje de error.	El sistema



		<p>presente, Eventos activo, ioc activo</p> <p>BD activa, Curso académico ausente, Eventos activo, ioc activo.</p> <p>BD activa, Curso académico presente, Eventos con problemas, ioc activo.</p> <p>BD activa, Curso académico presente, Eventos activo, ioc con problemas.</p> <p>BD activa, Curso académico presente, Eventos activo, ioc vacío.</p>		no muestra los Cursos académicos existentes.
		BD activa, Curso académico vacío, Eventos activo, ioc activo.	El sistema muestra un listado vacío.	El sistema al no obtener ningún dato muestra un listado vacío.
Se guarda el curso académico como información del evento.	Se selecciona el curso académico.	BD activa, Curso académico presente, Eventos activo, ioc activo.	El sistema almacena el identificador del curso académico seleccionado como información de la tabla correspondiente al evento.	El usuario selecciona el curso académico. El sistema almacena el identificador del curso académico en la base de datos como información correspondiente al evento.



No se guarda el curso académico seleccionado como dato del evento.	Se selecciona el curso académico.	BD con problemas, Curso académico presente, Eventos activo, ioc activo.	El sistema muestra un mensaje de error.	Una vez seleccionado el curso académico, el usuario escoge la opción aceptar. El sistema no guarda el curso académico, muestra un mensaje de error.
		BD activa, Curso académico presente, Eventos con problemas, ioc activo.		
		BD activa, Curso académico presente, Eventos activo, ioc con problemas.		

