Sistema de Gestión de Eventos Investigativos para la Universidad de las Ciencias Informáticas

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

Autor(es): Edelso Alejandro Rojas Villacampa
Jorge Miguel Soldevila Mustelier

Tutor(es): Ing. Yudanis Gago Martínez

Universidad de las Ciencias Informáticas
La Habana, 24 de junio del 2014


"La sabiduría suprema es tener sueños bastante grandes para no perderlos de vista mientras se persiguen."
William Faulkner

Declaración de autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) a que haga el uso que estimen pertinente con el mismo.

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

Autor:

_____ _____

Edelso Alejandro Rojas Villacampa Jorge Miguel Soldevila Mustelier

Tutor:

_____

Ing. Yudanis Gago Martínez

Agradecimientos

De Edelso:

A mi madre, porque todo lo que soy hoy, se lo debo a ella, por los 24 años de su vida que me regalo y los que está por regalarme, porque cambio su figura y su juventud por una barriga, porque cambio su delineador de ojos, por ojeras, porque cambio toda su vida sin exigir nada a cambio, a ella le regalo este día porque es quien más se lo merece.
A mi padre, quien me ha enseñado que lo importante no es cuantos errores hayas cometido sino tener la fuerza para arreglar cada uno de ellos, por todo lo que significas para mí, porque aunque no lo diga, estoy orgulloso de ser tu hijo y nada en el mundo va a cambiar eso.
A mi abuela del alma, mi segunda mamá, la que me tapaba todas mis travesuras de niño, la persona que solo tenía amor para darme y que vive por los ojos de este nieto que solo le da dolores de cabeza, que aun cuando la vida le daba golpes duros, ella siempre se mantenía firme, de pie, para ser el apoyo que toda la familia necesitaba.
A mi abuelo, a mi eterno abuelo, la persona que me dio posiblemente la lección más importante de todas, que cuando hay amor de verdad, la sangre no es importante, que aunque siento que la vida me lo quito demasiado pronto, donde quiera que estés, sé que te sientes orgulloso hoy de tu primer nieto.
A mis hermanas: Por ser los regalos más importantes que me han dado mis padres, porque se siempre estarán ahí para mí, quiero decirles que las amo y que aunque haga cosas que indiquen lo contrario, es porque soy medio estúpido y a veces no se me comportar pero nunca las dejare de amar.
A mi tío Raymel: por ser mi hermano mayor, porque aunque lo más que me da es trabajo, estoy muy agradecido de tenerlo en mi vida.
A mis tíos (Los Rojas): Por ser la gran familia que son, por acogerme como un hijo cada uno de ellos, y aunque somos muchos sobrinos, siempre tienen un momento para darte el cariño y el apoyo que necesitas, porque son la familia más unida que he tenido la posibilidad de conocer, y doy gracias por formar parte de ella.
A mi amigo Soto porque desde que tengo memoria siempre está ahí a mi lado, porque sé que puedo contar contigo, por ser un verdadero amigo.
A mis amigos: Omar, Roanni, Alex, Pedro, Yoandri, Marcos, Lisbeth, Darlenis porque ustedes son la verdadera justificación por la cual estoy hoy aquí, por convertirse en mis amigos, por demostrarme que están ahí para cualquier cosa, por ser mi apoyo cuando las cosas no salían bien, por ser los culpables de que este día no sea del todo feliz, por convertirse sin darme cuenta en mi familia, siempre estarán en mi corazón.
A mis profesores, por todo lo que hicieron y lo que no hicieron, porque fue lo que me condujo a este día.

A mis grupos, a todos los que formaron parte de cada grupo por el cual transite, por sopórtame todos los días, y hacer cada turno de clase, inolvidable.

De Jorge Miguel: Agradecer a todas las personas, profesores, recién graduados y estudiantes; que de una forma u otra contribuyeron a que se realizara mi mayor sueño, hablo de Nersa, Diego, Amado, Gustavo y Alejandro. No puedo dejar de mencionar un pequeño grupo que son como mi familia, ellos han estado en las buenas y malas apoyándome y regañándome también; les hablo de mis hermanos Eric, Carlos Ernesto (chino) y Osmel. Un saludo grande para el equipo principal de futsal de la facultad 4(CAMPEONES) pero mucho más grande para mi gente de LA FURIA; hemos hecho historia en todos estos años. Para el personal del festival que hemos pasados momentos amargos y muy buenos también; a las bailarinas principales Nersa, Aimé (micha) y Lisbeth (la gorda). Agradecer a una persona muy especial para mí, mi novia Yanelis por estar conmigo en todo momento, quisiera que fuera así por mucho tiempo.

Dedicatoria

De Edelso:
A mi madre, porque todo lo que soy se lo debo a ella, por su amor, por su paciencia, por su confianza, porque me apoyó, y me seguirá apoyando toda la vida en cualquier decisión errónea que sea capaz de tomar, porque aun estando lejos, encuentra la manera de estar presente a cada minuto de mi vida, por eso y muchas cosas más, te adoro mamita.

De Jorge Miguel:
A mi familia, principalmente a mis padres, este también es su triunfo, gracias por ser todo para mí, por darme más que amor y cariño sin ustedes no hubiera sido posible este sueño; por eso y mucho más esta tesis está dedicada a ustedes; las personas que más quiero.

Resumen

En la Universidad de las Ciencias Informáticas el proceso de gestión de eventos investigativos se hace en gran medida de forma manual, lo que lo convierte en un proceso muy complejo, debido a la gran cantidad de información que se debe manejar, además de no presentar una divulgación adecuada. El presente trabajo consiste en un sistema desarrollado con tecnologías libres para facilitar este proceso.

El desarrollo fue guiado por la metodología de desarrollo Programación Extrema. Como lenguajes de programación del lado del cliente se utilizaron HTML, JavaScript a través de la librería JQuery, CSS, y la tecnología AJAX. Como lenguaje del lado del servido se usó PHP a través del Sistema de Gestión de Contenido (CMS por sus siglas en inglés). Se usó el NetBeans como Entorno de Desarrollo Integrado, Apache como servidor web y PostgresSQL como gestor de bases de datos. Se implementaron 5 módulos: Comisión, Entity_example, Trabajo_evento, Trabajos_premiados, Temáticas, y se espera que se realicen mejoras futuras además de la incorporación de nuevos módulos. Fueron realizadas las pruebas de software a la propuesta de solución, las cuales garantizan el correcto funcionamiento del sistema y la satisfacción del cliente.

Índice

Introducción

Las Tecnologías de la informatización y las comunicaciones (TIC), a nivel mundial, han venido a fomentar la informatización de la sociedad y con ella los procesos que se manejan a nivel institucional, permitiendo elevar considerablemente la calidad de eventos que se desarrollan como parte del quehacer cotidiano de las distintas instituciones. Un Evento no es más que un suceso importante y programado, de índole social, académica, artística o deportiva. (RAE, 2001) En la actualidad, los eventos en todo el mundo son una fuente de preparación y superación para todos los participantes, se presentan como una de las vías más importantes de elevar el nivel cultural, deportivo o intelectual de los involucrados.

En Cuba, un país subdesarrollado, las TIC han jugado un papel fundamental en la gestión de eventos de cualquier índole, fomentando su uso en esferas que son priorizadas por el estado tales como la salud, el deporte y la educación, brindando las herramientas y tecnologías necesarias para la informatización de estos eventos y por consiguiente contribuyendo al desarrollo y calidad de los procesos asociados a cualquier institución.

Como parte de los procesos asociados al área de la educación se encuentra la gestión de eventos investigativos, proceso de vital importancia para cualquier institución educativa. En la Universidad de las Ciencias Informáticas (UCI) casa de altos estudios en Cuba, se desarrollan numerosos eventos de esta índole, que tienen como objetivo elevar el potencial investigativo del centro. Actualmente, concentrar la información que generan los diversos eventos investigativos desarrollados en este centro educacional constituye un problema. No se cuenta muchas veces con la información asociada a eventos pasados ya que al manejarla en formato duro en muchas ocasiones se pierde o deteriora con el pasar del tiempo; otras veces porque aunque se maneje en formato digital no existe organización ni centralización de la información que generan estos eventos, lo cual atenta contra el fácil acceso a la información, dificultando la búsqueda y análisis de resultados de investigaciones. No existe un mecanismo eficiente para brindar información detallada acerca de los distintos eventos investigativos, dígase fechas de celebración de los eventos, temáticas sobre las que se pueden escribir artículos, retroalimentación sobre el estado de un artículo una vez enviado para participar en un evento, constitución de la comisión en la que sesionará el trabajo, entre otros aspectos importantes que elevarían la calidad de estos espacios investigativos, además de provocar un aumento considerable en la participación por parte de estudiantes y trabajadores de este centro en eventos de este tipo.

Por tanto el **problema** a resolver puede quedar definido a modo de interrogante de la siguiente forma: **¿Cómo automatizar el proceso de gestión de eventos en la Universidad de las Ciencias Informáticas para lograr una centralización y organización de la información que generan estos eventos?**

El **objeto de estudio** fijado es el proceso de gestión de eventos, y el campo de acción en que se enmarca el trabajo es el proceso de gestión de eventos investigativos en la UCI.

El **objetivo general** de este trabajo es: Diseñar e implementar una herramienta informática que automatice los procesos asociados a los eventos investigativos celebrados en la UCI.

De una manera más detallada, los **objetivos específicos** para la construcción del sistema son:

- Investigar las tendencias tecnológicas actuales y estándares más adoptados en la construcción de sistemas de gestión de información a nivel mundial.
- Analizar, definir, estructurar y detallar mecanismos de diseño que garanticen el cumplimiento de las necesidades del sistema de software.
- Implementar el sistema diseñado haciendo uso de patrones y buenas prácticas de programación.
- Realizar pruebas al sistema durante su construcción.

Como **novedad práctica** se tiene que: Por primera vez la UCI contará con un sistema que gestione toda la información referente a los eventos investigativos celebrados en esta institución.

La **hipótesis** queda respaldada por la siguiente afirmación: Si se implementa un sistema para la gestión de eventos investigativos en la UCI, se reducirá en tiempo y esfuerzo la organización, además de automatizar el proceso de búsqueda para facilitar el análisis de resultados históricos.

Métodos investigativos:

Para llevar a cabo la investigación, se emplean los métodos de nivel teórico y empírico.

Métodos teóricos:

Análisis Histórico-Lógico: En la presente investigación se utiliza este método para realizar el estudio del estado del arte, o sea para investigar acerca de otras aplicaciones o soluciones similares y de los lenguajes y metodologías de desarrollo de software existentes, describir la metodología, herramientas y lenguaje a utilizar en el análisis, diseño e implementación del sistema.

Modelación: El uso de este método permitirá realizar los diagramas de clases de diseño, el modelo de datos y la distribución física del sistema.

Métodos empíricos:

Observación: Este método es el instrumento que permite estudiar más de cerca el objeto de la investigación, las acciones, causas, consecuencias, etc.

Entrevista: Para obtener información del cliente sobre los requisitos funcionales y no funcionales con los que debe contar el sistema.

Para una mejor comprensión de la investigación, cuyo diseño metodológico se acaba de describir, se decidió definir una estructura capitular que aporte cierto grado de organización y facilite el estudio del documento. Los capítulos que la conforman, son los siguientes:

Capítulo 1: Fundamentación Teórica

En este capítulo se presentan los elementos teóricos que fundamentan el sistema para la gestión de eventos investigativos. Se incluye un estudio del estado del arte de las similares, tanto en el plano nacional como internacional, además de incluir una investigación sobre las tecnologías y herramientas actuales para el desarrollo de aplicaciones web y la fundamentación de la selección de las más adecuadas para ser usadas en el desarrollo de la solución al problema actual.

Capítulo 2: Exploración y Planificación

Este capítulo está dedicado a las dos primeras fases de la metodología de desarrollo XP (Programación Extrema), escogida para guiar el proceso de desarrollo, definidas para la investigación: exploración y planificación. En la exploración los clientes plantean a grandes rasgos historias de usuarios que son de interés para la primera entrega del producto, el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Durante la planificación el cliente establece la prioridad de cada historia de usuario y los programadores realizan una estimación del esfuerzo para implementarlas, se realiza el Plan de Iteraciones y el Plan de Entregas del producto. El éxito del proyecto depende en gran medida de las decisiones tomadas en estas fases.

Capítulo 3: Implementación y Pruebas

Este capítulo está dedicado a la tercera y cuarta fase de la metodología de desarrollo XP definidas para la investigación: implementación y pruebas. Como parte de la fase implementación, en la investigación se presenta una descripción de la arquitectura de la propuesta de solución, se presentan

las tarjetas CRC (Clase-Responsabilidades-Colaboradores) y el modelo de datos a través de un DER (Diagrama Entidad-Relación). Se realiza una descripción de los patrones de diseño, los estilos y estándares de codificación utilizados para implementar la solución. También se muestran las tareas de ingeniería necesarias para llevar a cabo el proceso de desarrollo. En la fase de pruebas se define la estrategia de pruebas a seguir, y se realiza las pruebas de aceptación.

Capítulo 1: Fundamentación Teórica.

1.1 Los sistemas de Gestión de Información.

En la era de la información, de la explosión de sus tecnologías, se vive la etapa en la que la humanidad ha alcanzado un desarrollo imprevisible; cada día son mayores las diferencias sociales, políticas y económicas. Se habla constantemente sobre la sociedad de la información, es visible el paso de las sociedades industriales a las posindustriales y del conocimiento, donde el factor esencial de progreso es el conocimiento. Esta nueva sociedad, con organizaciones basadas en el aprendizaje, cuyo capital más preciado es el ser humano, se sustenta en un desarrollo tecnológico sin precedentes, es el punto en el cual las grandes compañías planifican sus productos en función de la gestión del conocimiento y de la viabilidad para su obtención. En este contexto, debe entenderse que las tecnologías de información y las telecomunicaciones no son más que un medio para transmitir y gestionar datos, información y conocimiento, el conocimiento es factor fundamental para la creación de riquezas. (Acimed, 2002)

En el mundo de la Tecnología Web la informatización de procesos ha ido evolucionando, convirtiéndose en algo de orden primario en el desarrollo de las TIC. Las mejores empresas que operan en el siglo XXI se enfrentan a muchos retos significativos:

- Rentabilidad.
- Competitividad.
- Globalización.
- Velocidad de los cambios.
- Capacidad de adaptación.
- Crecimiento.
- Tecnología

Equilibrar estos y otros requisitos empresariales puede constituir un proceso difícil y desalentador. Es aquí donde entran en juego los sistemas de gestión, al permitir aprovechar y desarrollar el potencial existente en la organización.

La implementación de un sistema de gestión eficaz puede ayudar a:

- Gestionar los riesgos sociales, medioambientales y financieros.
- Mejorar la efectividad operativa.
- Reducir costos.
- Aumentar la satisfacción de clientes y partes interesadas.

- Proteger la marca y la reputación.
- Lograr mejoras continuas.
- Potenciar la innovación.
- Eliminar las barreras al comercio.
- Aportar claridad al mercado.

1.2 Estudio de sistemas similares.

En este acápite se realizara un estudio detallado de alguna de las herramientas más destacadas en el ámbito de gestión de eventos, tanto internacional como nacional, el cual ayudará a entender mejor su función, además de las tendencias y metodologías usadas en su proceso de desarrollo, el detallado estudio de estos sistemas también ayudaran a incorporar a la presente solución varias funcionalidades importante para la obtención de un producto que cumpla las exigencias del cliente.

1.2.1 Sistemas a nivel Internacional.

En el plano internacional cada vez son más populares los sistemas de gestión de eventos, para gestionar toda la información referente a estos. Estos sistemas son concebidos bajo una serie de conceptos de acuerdo al ámbito en que se vean relacionados, caracterizándose siempre por estar diseñados para satisfacer las necesidades de los usuarios, exponen algunos principios básicos de la web 2.0 como la interoperabilidad[1], el diseño centrado en el usuario[2] y la inclusión de la tecnología móvil para la visualización de sistemas mediante vistas específicas. Este estudio estuvo centrado en la necesidad de recopilar datos acerca de las funcionalidades básicas, que presentan sistemas de este tipo, importantes a tener en cuenta para la realización de la presente solución. Entre los distintos sistemas existentes se estudió por su estructura y el uso de estándares SYM.POSIUM, de la Universidad de Murcia (España), y el Sistema de Gestión de Eventos de la Universidad de Icesi (Colombia)

SYM.POSIUM: Es una herramienta potente, que ofrece soluciones para la organización y gestión de eventos, pero también flexible, adaptándose a eventos más pequeños y también a las necesidades de los asistentes. En su página principal se listan los últimos eventos publicados, o sea, los que están en curso, y los próximos eventos. Brinda, entre sus funcionalidades más importantes, la de ¨Buscar¨ eventos, artículos de interés, entre otros. (Sym.posium, 2014)

[1] *Habilidad de dos o más sistemas o componentes para intercambiar información y utilizar la información intercambiada.*
[2] Filosofía de diseño que tiene por objetivo la creación de productos que resuelvan necesidades concretas de sus usuarios finales.

Entre sus funcionalidades se encuentran:

- Dar de alta el evento e introducir los datos básicos.
- Gestionar las inscripciones de asistentes.
- Personalizar el formulario de inscripción.
- Definir el organizador y la categoría del evento.
- Crear las páginas de programa y ponentes.
- Introducir patrocinadores.
- Crear nuevas páginas.
- Introducir noticias.
- Publicar archivos.
- Enviar correos a asistentes.

Esta herramienta permite a los usuarios registrados tanto crear un evento, como gestionar su participación, además dispone de conexión con redes sociales para informar a colegas de la actividad de usuarios del sistema. De esta forma se favorece la difusión del evento al que asiste un usuario y además permite de esta forma actualizar la web, blog o red social del usuario.

Sym.posium también ofrece servicios de gran utilidad para los participantes de un evento. Se puede gestionar y difundir la actividad, así como contactar de forma fácil con otros asistentes y hacer trabajos colaborativos en la red (*networking*). (Sym.posium, 2014)

Sistema de Gestión de Eventos de la Universidad de Icesi: Es una herramienta que gestiona los eventos organizados en esta institución de altos estudios. Este sistema cuenta con una interfaz de usuario diseñada para facilitar el entendimiento del mismo, en su página principal se encuentra una lista de los eventos organizados. Cuenta con una serie de funcionalidades como son: (Icesi, 2014)

- Ingresar al sistema.
- Gestionar cuenta.
- Selección de idiomas.
- Sistema de Búsqueda.
- Sistema de ayuda.

Este sistema procesa mucha información de los distintos eventos, como puede ser la fecha de inicio y fin, los organizadores, posibles invitados, cronograma con las actividades, listado con las normas que deben cumplir los trabajos para concursar, además de contar, de acuerdo a su organización con: (Icesi, 2014)

- *Fanspage*: Página de seguimiento del evento en las redes sociales.
- Pre-inscripción: Algunos eventos brindan la posibilidad de inscribirte en el evento sin la necesidad de participar, permitiendo consultar toda la bibliografía o artículos publicados.
- Sitio web oficial: Algunos eventos son patrocinados por instituciones que cuentan con páginas web propias y la utilizan para la divulgación de información, etc.

1.2.2 Sistemas a nivel nacional.

En Cuba, se ha venido trabajando para implementar estos sistemas por las facilidades que brindan en el ámbito académico. Algunas de las universidades más importantes cuentan con sistemas informáticos que aunque no sean los más óptimos, ponen a disposición de los usuarios una serie de funcionalidades que favorecen los procesos de divulgación y participación de los distintos eventos que se celebran.

Instituto Superior Politécnico "José Antonio Echeverría": Cuenta con un sistema de gestión de eventos integrado a su sitio web principal en el cual muestra una serie de eventos en línea que son organizados en la propia institución, además de mostrar eventos externos y eventos pasados. Cada uno de los eventos listados presenta una serie de elementos asociados como son: (CUJAE, 2014)

- Presentación: Se realiza un breve estudio del tema del evento, además de los patrocinadores del mismo.
- Temáticas: Se desglosan las diferentes temáticas con las que cuenta el evento.
- Comités: Se muestran los diferentes comités que participan en el evento.
- Calendario: Se muestra un calendario con el programa técnico general, además de mostrar otro calendario con las conferencias magistrales programadas.
- Se muestra un listado con todas las ponencias recibidas especificando:
  - ✓ Código del trabajo.
  - ✓ Título del trabajo.
  - ✓ Nombre del autor.
  - ✓ País del autor.

Universidad de la Habana: Cuenta con un calendario en su sitio web principal donde se muestran todos los eventos anunciados con su respectiva fecha, donde cada evento trae consigo un documento oficial con una breve descripción propia, todos los comités participantes, los organizadores por cada institución participante, un calendario con el programa general, y por último, enumeran todas las actividades programadas para el evento en cuestión teniendo en cuenta desde ponencias hasta talleres asociados. (Habana, 2014)

Serie Científica: La Universidad de las Ciencias Informáticas (UCI) cuenta con una herramienta que gestiona diferentes artículos investigativos tanto de profesores como alumnos destacados en las diferentes esferas científicas de la institución.

Brinda una serie de informaciones muy útiles como son las normas de publicación en la revista, en la cual se definen las pautas para cada publicación, así como los estándares con los que debe cumplir un artículo para que sea válido, brinda la posibilidad de buscar artículos filtrando por: (Científica, 2014)

- Autor.
- Título.
- Palabras claves.

Además, posibilita suscribirse para recibir notificaciones, vía correos electrónicos, de los últimos artículos y publicaciones gestionados.

Los artículos publicados son validados por un Comité Científico especializado en el tema en cuestión, también del Consejo Editorial de la Serie Científica. (Científica, 2014) Este sistema forma parte del estudio debido a la existencia de similitudes en el sistema de gestión de la información, al manejar aspectos importantes como la publicación de artículos y la validación de estos por una comisión especializada en el tema, además de almacenar una gran cantidad de información sobre diversos temas para futuras consultas.

En general, los sistemas de gestión de eventos son herramientas creadas para facilitar el proceso de organización, gestión, y divulgación de los eventos institucionales de cualquier índole, los sistemas estudiados presentan similitudes en cuanto a su estructura, como son:

- Cuentan con un calendario con los eventos organizados.
- Cuentan con un sistema de búsqueda.
- Brindan la posibilidad de gestionar el perfil de los usuarios.
- Brindan servicio de notificación vía email.

Estas herramientas estudiadas anteriormente no pueden ser tomadas como solución al problema de la presente investigación porque, en algunos casos, son sistemas privativos costosos, además de no adaptarse totalmente a las especificaciones requeridas, aunque presentan características que serán tomadas en cuenta para la realización del sistema que se propone como resultado del presente trabajo, además de incorporarle nuevas funcionalidades como la gestión de notificaciones que posibiliten una mejor gestión de los distintos eventos desarrollados en la universidad, atendiendo a las características propias de cada uno de ellos.

1.3 Estudio de las metodologías y estándares para el desarrollo del software.

Al transcurrir los primeros años de la década de los 70 el proceso de desarrollo de software a nivel mundial se encontraba envuelto en una crisis, desatada por malas prácticas empleadas en el proceso de desarrollo de software. Esta situación provocó que comenzaran a surgir estándares y metodologías que estructuran un desarrollo de software más organizado y con mejor calidad. Surge así una nueva disciplina, la Ingeniería de Software que según unos de sus padres, es “Una disciplina que integra el proceso, los métodos, y las herramientas para el desarrollo de software de computadora.” (Pressman, 2002)

Precisamente una de las buenas prácticas que promueve esta disciplina es el uso de metodologías para guiar el proceso desarrollo de software. A nivel internacional, las instituciones y empresas dedicadas a la industria de software, emplean en su actividad de desarrollo, modelos, metodologías o procedimientos estándares para desarrollar, instalar y mantener un producto de este tipo.

El objetivo de un proceso de desarrollo de este tipo, es elevar la calidad del software (en todas las fases por las que pasa) a través de una mayor transparencia y control sobre el proceso. Da igual el alcance que se desee, hay que producir lo esperado en el tiempo y con el costo esperado.

Es labor del proceso de desarrollo hacer que esas medidas para aumentar la calidad sean reproducibles en cada desarrollo. Estas metodologías establecen un conjunto de actividades que definen cómo se debe hacer el software, quién debe hacer cada actividad, cuándo hacerla y qué se debe hacer.

La implantación de un proceso de desarrollo es una labor más a medio-largo plazo que una labor de resultados inmediatos. Cuesta tiempo que los trabajadores se adapten al proceso, pero una vez superado, la inversión se recupera con creces. Es por ello que no tiene sentido ajustarse a un proceso al pie de la letra, sino que hay que adaptarlo a las necesidades y características de cada empresa, equipo de trabajo o casi a cada proyecto. (Molpeceres, 2002)

Actualmente existe una gran cantidad de procesos de desarrollo, agrupados en dos tendencias: los métodos ágiles y los métodos pesados. La diferencia fundamental entre ambos es que mientras los métodos pesados intentan conseguir el objetivo común por medio de orden y documentación, los métodos ligeros (también denominados métodos ágiles), tratan de mejorar la calidad del software por medio de una comunicación directa e inmediata entre las personas que intervienen en el proceso.

1.3.1 Métodos Pesados. *Rational Unified Process* (RUP)

RUP es uno de los procesos más generales que existen actualmente, su finalidad no está restringida a guiar desarrollo de software, sino cualquier tipo de proyecto. La estrategia de este proceso es conseguir su objetivo por medio de orden y documentación, lo que lo convierte en el más fiel exponente de los métodos pesados. RUP define cuatro fases (inicio, elaboración, construcción y transición) y dentro de cada una de ellas el equipo de trabajo pasa por todos los flujos que son transversales a las fases, inclusive en varias iteraciones (ver Anexo 2). (Kruchten, 2003)

Fases del proceso:

1. Inicio: Se describe el negocio y se delimita el proyecto, describiendo sus alcances con la identificación de los casos de uso del sistema.
2. Elaboración: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen. A pesar de que se desarrolla a profundidad una parte del sistema, las decisiones sobre la arquitectura se hacen sobre la base de la comprensión del sistema completo y los requerimientos (funcionales y no funcionales) identificados de acuerdo con el alcance definido.
3. Construcción: Se logra un producto listo para su utilización que está documentado y tiene un manual de usuario. Se obtiene una o varias versiones del producto que han pasado las pruebas. Se ponen estos entregables a consideración de un subconjunto de usuarios.
4. Transición: La versión ya está lista para su instalación en las condiciones reales. Puede implicar reparación de errores.

Flujos de trabajo:

1. Modelado del negocio: Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
2. Requerimientos: Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
3. Análisis y diseño: Describe cómo el sistema será realizado a partir de la funcionalidad prevista y las restricciones impuestas (requerimientos), por lo que indica con precisión lo que se debe programar.
4. Implementación: Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación.
5. Pruebas: Busca los defectos a lo largo del ciclo de vida.

6. Despliegue: Produce un entregable del producto y realiza actividades como empaque, instalación, asistencia a usuarios, entre otros. Para entregar el software a los usuarios finales.
7. Gestión de proyectos: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
8. Gestión del cambio y configuraciones: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos, control de versiones, etc.
9. Entorno: Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización.

Las principales bibliografías que abordan esta metodología coinciden en definir tres características fundamentales al hablar de RUP:

a. Dirigido por casos de uso: Los casos de uso (CU) reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo, ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso.
b. Centrado en la arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente. RUP se desarrolla mediante iteraciones, comenzando por los CU relevantes desde el punto de vista de la arquitectura. El modelo de arquitectura se representa a través de vistas, o sea perspectivas del sistema, que logran una abstracción particular en cada uno de los casos, en otras palabras, se trata de que en cada una de las llamadas vistas se represente el sistema en su totalidad teniendo en cuenta solo determinados aspectos.
c. Iterativo e incremental: RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Las iteraciones hacen referencia a pasos en los flujos de trabajo, y los incrementos, al crecimiento del producto.

RUP es más adecuado para proyectos grandes, dado que requiere un equipo de trabajo capaz de administrar un proceso complejo en varias etapas; en proyectos pequeños es posible que no se pueda cubrir los costos de dedicación del equipo de profesionales necesarios. Los requerimientos de los

diversos inversores pueden ser diferentes, contradictorios o disputarse recursos limitados, así que debe encontrarse un balance que satisfaga los deseos de todos y el control de calidad no debe realizarse al final de cada iteración, sino en todos los aspectos de la producción. (Kruchten, 2003)

1.3.2 Desarrollo Ágil. *Extreme Programming* (XP)

XP se clasifica como una metodología ágil, que se enfoca en potenciar las relaciones entre los miembros del equipo de desarrollo como la clave para el éxito, promueve el trabajo en equipo, se interesa por la superación de los desarrolladores, y favorece un buen clima de trabajo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, en la comunicación fluida entre todos los participantes, la sencillez de las soluciones desarrolladas y en la audacia para enfrentar los cambios que surjan. Por lo que XP se adecúa perfectamente a proyectos con requisitos imprecisos y propensos al cambio. (Letelier, y otros, 2006)

Las metodologías de desarrollo de software denominadas ágiles se sustentan sobre la base de los valores y principios fijados en el Manifiesto Ágil. (Agile manifesto, 2001)

Principales valores del desarrollo ágil:

- Al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas.
- Desarrollar software que funcione más que conseguir una buena documentación.
- La colaboración con el cliente más que la negociación de un contrato.
- Responder a los cambios más que seguir estrictamente un plan.

Principios del desarrollo ágil:

- La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor.
- Dar la bienvenida a los cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva.
- Entregar frecuentemente software que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.
- El cliente y los desarrolladores deben trabajar juntos a lo largo del proyecto.
- Construir el proyecto en torno a individuos motivados.
- El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.
- El software que funciona es la medida principal de progreso.
- Los procesos ágiles promueven un desarrollo sostenible.

- La atención continua a la calidad técnica y al buen diseño mejora la agilidad.
- La simplicidad es esencial.
- Las mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos.
- En intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivo, y según esto ajusta su comportamiento.

La **Tabla 1**(ver Anexo 1) compara las distintas aproximaciones ágiles en base a tres parámetros: vista del sistema como algo cambiante, tener en cuenta la colaboración entre los miembros del equipo y características más específicas de la propia metodología como son simplicidad, excelencia técnica, resultados, adaptabilidad y prácticas de colaboración. También incorpora como referencia no ágil el *Capability Madurity Model* (CMM). (Highsmith, 2002)

Dentro de las metodologías ágiles, XP clasifica como una de las que más seguidores poseen, y la que mejor se ajusta a cualquier ambiente de desarrollo ágil.

XP propone los siguientes roles para un equipo de desarrollo: (Beck, 2008)

- **Programador**: El programador escribe las pruebas unitarias y produce el código del sistema. Debe existir una comunicación y coordinación adecuada entre los programadores y otros miembros del equipo.
- **Cliente**: El cliente escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio. El cliente es solo uno dentro del proyecto, pero puede corresponder a un interlocutor que está representando a varias personas que se verán afectadas por el sistema.
- **Encargado de pruebas (*Tester*)**: El encargado de pruebas ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.
- **Encargado de seguimiento (*Tracker*)**: El encargado de seguimiento proporciona retroalimentación al equipo. Su responsabilidad es verificar el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, comunicando los resultados para mejorar futuras estimaciones. También realiza el seguimiento del progreso de cada iteración y evalúa si los objetivos son alcanzables con las restricciones de tiempo y recursos presentes. Determina cuándo es necesario realizar algún cambio para lograr los objetivos de cada iteración.

- **Entrenador (*Coach*)**: Es responsable del proceso global. Es necesario que conozca a fondo el proceso XP para proveer guías a los miembros del equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente.
- **Consultor**: Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto. Guía al equipo para resolver un problema específico.
- **Gestor *(Big boss*)**: Es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es de coordinación.

Un proyecto XP tiene éxito cuando el cliente selecciona el valor de negocio a implementar basado en la habilidad del equipo para medir la funcionalidad que puede entregar a través del tiempo. El ciclo de desarrollo consiste (a grandes rasgos) en **los siguientes pasos**:

- El cliente define el valor de negocio a implementar.
- El programador estima el esfuerzo necesario para su implementación.
- El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
- El programador construye ese valor de negocio.
- Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración. (Letelier, y otros, 2006)

El ciclo de vida ideal de XP consiste de seis fases: Exploración, Planificación de la Entrega (*Release*), Iteraciones, Producción, Mantenimiento y Muerte del Proyecto. (Beck, 2008)

- **Fase I: Exploración**: En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.
- **Fase II: Planificación de la Entrega**: En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera

entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días. Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación.

- **Fase III: Iteraciones**: Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de Entrega está compuesto por iteraciones de no más de tres semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra escogiendo las historias que fuercen la creación de esta arquitectura, sin embargo, esto no siempre es posible ya que es el cliente quien decide qué historias se implementarán en cada iteración (para maximizar el valor de negocio). Al final de la última iteración el sistema estará listo para entrar en producción.
- **Fase IV: Producción**: La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase. Es posible que se rebaje el tiempo que toma cada iteración, de tres a una semana. Las ideas que han sido propuestas y las sugerencias son documentadas para su posterior implementación (por ejemplo, durante la fase de mantenimiento).
- **Fase V: Mantenimiento**: Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura.
- **Fase VI: Muerte del Proyecto**: Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo. La aplicación disciplinada de las prácticas que se describen a continuación, conlleva a un crecimiento lento del costo del cambio, con un comportamiento asintótico.

Ilustración 1. Ciclo de vida de XP

1.3.3 Análisis de la selección del Proceso de Desarrollo.

La elección de la metodología que guiará el proceso de desarrollo del software, se apoya fundamentalmente en la comparación resumen que se muestra en la Tabla 2. Teniendo en cuenta elementos esenciales como: Cantidad de Artefactos, cantidad de roles, ocurrencia de cambios, el cliente como parte de equipo de desarrollo, dimensión del proyecto, tamaño del equipo de desarrollo, período de tiempo para el desarrollo, entre otros. Evidenciándose la necesidad de adoptar una metodología ágil para el desarrollo del sistema, ya que se cuenta con un equipo pequeño (dos personas desarrolladores y el cliente), un proyecto pequeño, poco tiempo de desarrollo y los requisitos no se conocen con exactitud.

Dentro de las metodologías ágiles se decide emplear XP, atendiendo fundamentalmente a que es la que mejor se ajusta a cualquier ambiente de desarrollo ágil de los analizados, quedando esto evidenciado en la comparación que recoge la Tabla 1 antes presentada. A continuación se enumeran las principales diferencias de una Metodología Ágil respecto a las Metodologías Tradicionales (llamadas peyorativamente “no ágiles” o “pesadas”).

La Tabla 2 recoge estas diferencias que no se refieren solo al proceso en sí, sino también al contexto de equipo y organización que es más favorable a cada uno de estas filosofías de procesos de desarrollo de software. (Letelier, y otros, 2006)

Metodología Ágil	Metodología Tradicional
Pocos Artefactos. El modelado es prescindible, modelos desechables.	Más Artefactos. El modelado es esencial, mantenimiento de modelos

Pocos Roles, más genéricos y flexibles	Más Roles, más específicos
No existe un contrato tradicional, debe ser bastante flexible	Existe un contrato prefijado
Cliente es parte del equipo de desarrollo (además in-situ)	El cliente interactúa con el equipo de desarrollo mediante reuniones
Orientada a proyectos pequeños. Corta duración (o entregas frecuentes), equipos pequeños (< 10 integrantes) y trabajando en el mismo sitio	Aplicables a proyectos de cualquier tamaño, pero que suelen ser especialmente efectivas/usadas en proyectos grandes y con equipos posiblemente dispersos.
La arquitectura se va definiendo y mejorando a lo largo del proyecto	Se promueve que la arquitectura se defina tempranamente en el proyecto
Énfasis en los aspectos humanos: el individuo y el trabajo en equipo	Énfasis en la definición del proceso: roles, actividades y artefactos
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Se esperan cambios durante el proyecto	Se espera que no ocurran cambios de gran impacto durante el proyecto

Tabla#1. Diferencias entre metodologías ágiles y no ágiles.

1.4 Lenguaje de Modelado.

Principales características de UML[3]:

- Permite modelar sistemas utilizando técnicas orientadas a objetos (OO).
- Mediante UML se pueden especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.
- Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas, versiones, entre otros).
- Cubre las cuestiones relacionadas con el tamaño propio de los sistemas complejos y críticos.

[3] **Lenguaje Unificado de Modelado** (LUM o **UML**, por sus siglas en inglés, *Unified Modeling Language*)

- Es un lenguaje muy expresivo que cubre todas las vistas necesarias para desarrollar y luego desplegar los sistemas.

El Lenguaje Unificado de Modelado proporciona a los desarrolladores un vocabulario que incluye tres categorías: elementos, relaciones y diagramas. (Kruchten, 2003)

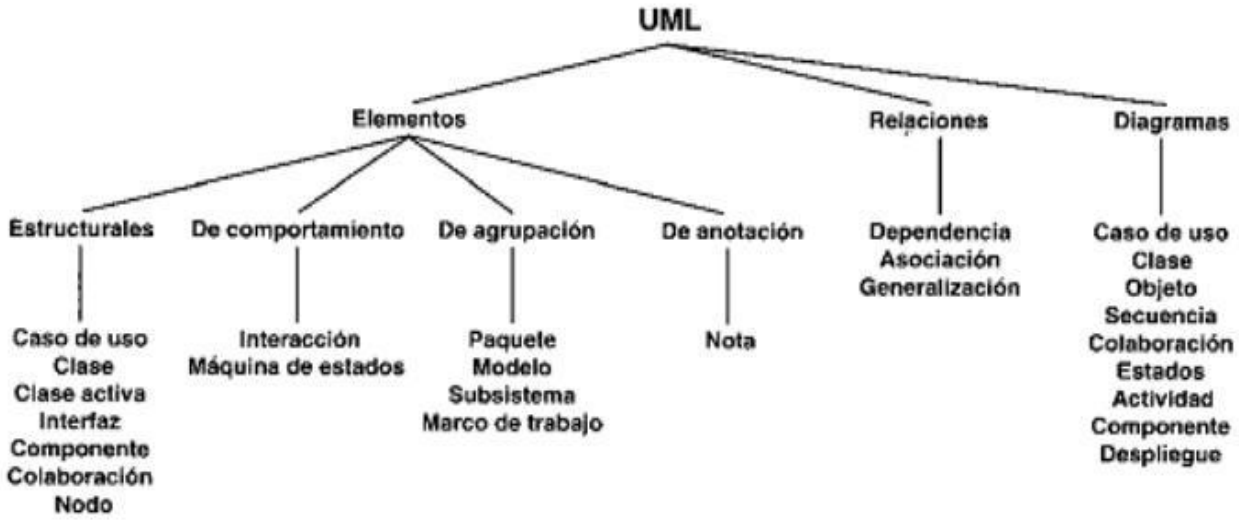

Ilustración 2. Vocabulario de UML.

1.4.1 Herramientas CASE[4] para el modelado UML

Desde la publicación oficial del UML a fines de 1997, la cantidad de las herramientas comerciales para el modelado con UML se incrementó dramáticamente. Esto provee de un mayor número de alternativas y exige realizar una investigación más profunda para seleccionar la herramienta de modelado UML que responda mejor a los requisitos del negocio y desarrollo de software de aplicación que permite lograr el mejor retorno de la inversión.

A medida que los sistemas que hoy se construyen se tornan más complejos, las herramientas de modelado con UML ofrecen muchos beneficios para todos los involucrados en un proyecto, por ejemplo, administrador del proyecto, analistas, arquitectos, desarrolladores y otros. Las herramientas CASE de modelado con UML permiten aplicar la metodología de análisis y diseño orientados a objetos y abstraerse del código fuente, en un nivel donde la arquitectura y el diseño se tornan más obvios y

[4] **CASE** corresponde a las iniciales de: *Computer Aided Software Engineering*; y en su traducción al Español significa Ingeniería de Software Asistida por Computación.

más fáciles de entender y modificar. Cuanto más grande es un proyecto, es más importante utilizar una herramienta CASE. Por otro lado, al usar las herramientas CASE:

- Los Analistas de Negocio/ Sistemas pueden capturar los requisitos con un modelo de casos de uso.
- Los Diseñadores/Arquitectos pueden producir el modelo de diseño para articular la interacción entre los objetos o los subsistemas de la misma o de diferentes capas (los diagramas UML típicos que se crean son los de clases y los de interacción).
- Los Desarrolladores pueden transformar rápidamente los modelos en una aplicación funcionando, y buscar un subconjunto de clases y métodos y asimilar el entendimiento de cómo lograr interfaces con ellos.

Entre los principales beneficios que ofrece la utilización de una herramienta CASE se encuentran:

- Mejora la comunicación entre usuario y especialista, ya que al tener incorporada la visualización de diagramas y de otras herramientas del análisis estructurado, actúa como un elemento que acelera la relación usuario/especialista.
- Permite la facilidad de revisión de aplicaciones instaladas.
- Son capaces de generar automáticamente las instrucciones del programa fuente. Esto permite acelerar el tiempo dedicado a la elaboración de programas y asegura además una estructura estándar y consistente para el programa.
- Da la posibilidad de la generación de documentación técnica.
- Dentro de la gama de herramientas de modelado utilizadas en el mundo se encuentran *Rational Rose* y *Visual Paradigm*, que constituyen dos de las herramientas más usadas en la actualidad.

1.4.2 *Visual Paradigm*

Es una herramienta diseñada para desarrollar software con programación orientada a objetos. Busca reducir la duración del ciclo de desarrollo, brindando ayuda a arquitectos, analistas, diseñadores y desarrolladores; permite el uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación y tiene capacidades de ingeniería directa e inversa. Usa UML como lenguaje de modelado. Presenta una interfaz de uso intuitiva y con muchas facilidades a la hora de modelar los diagramas que soportan la Ingeniería de Requerimientos. (Visual-Paradigm, 2014)

Una de las características más importantes de su uso es que brinda la posibilidad de sincronización del modelo de diseño y el código en todo el ciclo de desarrollo una vez que se integra con el Eclipse[5], permitiendo la facilidad de programar directamente sobre el código fuente generado y a su vez actualizar el diseño con cambios que se realicen en la programación. Tiene una alta disponibilidad de múltiples versiones para cada necesidad, al igual que en las plataformas Linux[6], MacOS[7] y Windows[8].

Presenta un innovador analizador textual donde se introduce texto extraído de conversaciones con el cliente, se definen actores, entidades, casos de uso disponibles para la generación de artefactos posteriores. Así mismo posee una herramienta de generación de reportes en formato PDF[9] o HTML[10] configurable y selectiva, se integra con entornos como Eclipse, Hibernate[11] y Subversion[12], e importa o exporta formatos estándares de otras herramientas CASE como el *Rational Rose*. (Visual-Paradigm, 2014)

1.4.3 *Rational Rose*

Es la herramienta CASE desarrollada por los creadores de UML (*Booch, Rumbaugh, Jacobson*), que cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases y entregables. El navegador UML de *Rational Rose* permite establecer una trazabilidad real entre el modelo (análisis y diseño) y el código ejecutable. Facilita el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información (vista de caso de uso, vista lógica, vista de componente y vista de despliegue), pero utilizan un lenguaje común para comprender y comunicar la estructura y la funcionalidad del sistema en construcción.

Rational Rose presenta un entorno de diseño muy atractivo y los cambios rápidos en el código a medida que se va desarrollando el modelado son muy buenos. Permite la generación de documentos en formato HTML, XML[13], y PDF de los reportes. (Pereda Ojeda, y otros, 2008)

[5] Entorno de desarrollo, comenzó como un proyecto de IBM Canadá. Fue desarrollado por OTI (*Object Technology International*).
[6] Sistema operativo libre homólogo a *Unix* que usualmente utiliza herramientas de *Sistema GNU.*
[7] Sistema operativo privativo muy popular en el mercado mundial.
[8] Sistemas operativo desarrollado por *Microsoft* desde 1981, año en que el proyecto se denominaba *«Interface Manager».*
[9] PDF (acrónimo del inglés *portable document format*, formato de documento portátil) es un formato de almacenamiento de documentos.
[10] HTML (*Hipertext Markup Language)* es el lenguaje de hipertexto con el que se escriben las páginas web.
[11] Herramienta de Mapeo Objeto-Relacional (ORM por sus siglas en inglés, *Object-Relational Mapping*) para la plataforma Java.
[12] Software para el control de versiones.
[13] XML (siglas en inglés de *eXtensible Markup Language*) es un lenguaje de marcado sencillo similar al HTML.

1.4.4 Análisis de la selección de la herramienta CASE.

El Visual Paradigm, en su versión 8.0, constituye un candidato más fuerte para su uso como herramienta CASE en la solución propuesta, debido principalmente a que iguala a *Rational Rose* en cuanto a capacidades y es mucho más integrable con el resto de las herramientas y estándares escogidos.

1.5 *Content Management System* (CMS).

Un sistema de gestión de contenidos (*Content Management System* en inglés, abreviado CMS), es un software que permite crear una estructura base para la creación y administración de contenidos, principalmente de páginas web.

Generalmente un CMS es una aplicación con una base de datos asociada en la que se almacenan los contenidos, separados de los estilos o diseño. El CMS controla también quién puede editar y visualizar los contenidos, convirtiéndose en una herramienta de gestión integral para la publicación de sitios web.

Algunas funcionalidades típicas de un CMS son**:**

- **Administración de la estructura del portal:** módulos, menús, diseño, configuración general, entre otros.
- **Administración del contenido:** distintos tipos de contenidos, gestión y publicación de contenidos, etc.
- **Administración de usuarios:** políticas de gestión de usuarios y de acceso a los contenidos mediante roles y permisos, entre otros.
- I**nformes y gestión del portal:** errores, estadísticas de acceso, etc.

De los CMS genéricos, algunos de los más utilizados en la actualidad son TYPO3[14], Joomla[15] y Drupal[16]. Todos ellos fueron publicados a principios de este siglo y tienen en común que están desarrollados en PHP[17] y MySQL[18] y que se distribuyen como software libre.

Cada uno de ellos cuenta con su propia comunidad de usuarios y desarrolladores que contribuyen al desarrollo del proyecto, ya sea trabajando en la mejora del software o aportando nuevos módulos para incrementar o mejorar sus funcionalidades. (Rodríguez, 2012)

[14] Marco de gestión de contenido, libre y de código abierto, que se distribuye bajo la Licencia Pública General de GNU.
[15] Sistema de gestión de contenido, de código abierto que se distribuye bajo una licencia GPL.
[16] Sistema de gestión de contenido modular y muy configurable, de código abierto, que se distribuye bajo la licencia de GNU/PL.
[17] Acrónimo recursivo que significa *PHP Hypertext Pre-processor* (inicialmente *PHP Tools*, o, *Personal Home Page Tools).*
[18] Es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones.

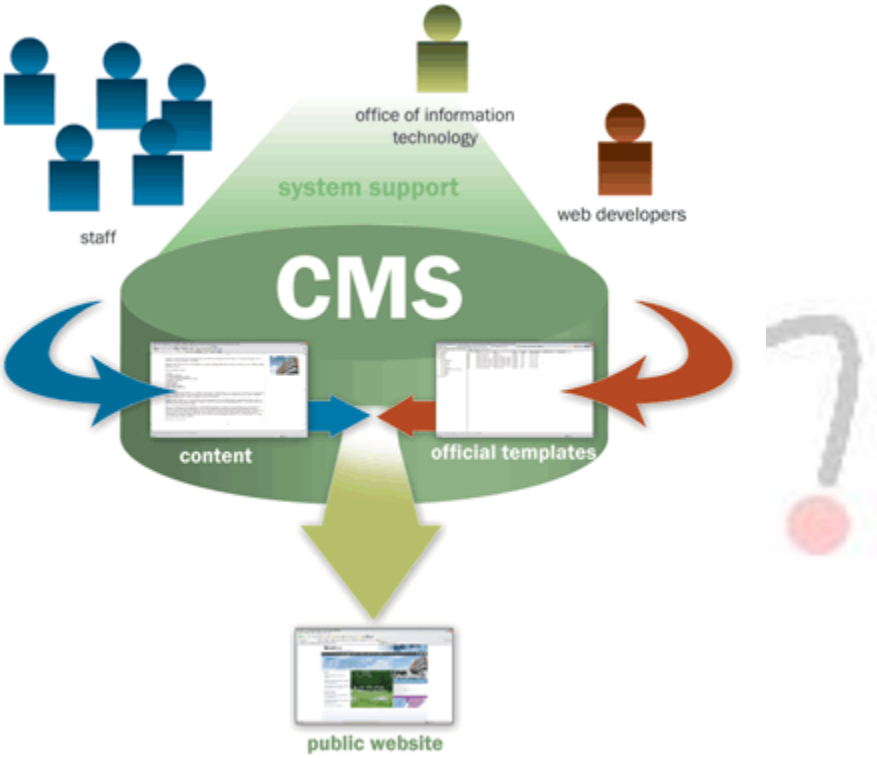

Ilustración 3. Funcionamiento de un CMS.

1.5.1 CMS Drupal

Drupal es un potentísimo sistema de gestión de contenidos, que ofrece más que probadas capacidades para la creación, desarrollo y mantenimiento de servicios y productos de información digital. Se encuentra expandido e instalado a nivel mundial, y ofrece gran cantidad de soluciones para todo tipo de contextos y problemas. Es un Sistema de gestión de contenidos (CMS) que se distribuye como software libre bajo licencia GNU GPL [19] versión 2 o superior. Puede ser modificado y distribuido libremente, pero siempre se debe hacer bajo la misma licencia. Esto quiere decir que si, por ejemplo, se desarrolla un módulo específico, este se debe distribuir con todos sus archivos fuente, de forma que cualquier otra persona pueda a su vez modificarlo y distribuirlo. (Tramullas, 2010)

El software está desarrollado con el lenguaje de programación PHP y utiliza una base de datos MySQL. Está maquetado con hojas de estilo CSS[20], con lo que es posible construir sitios web totalmente

[19] La Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General Public License o simplemente sus siglas del inglés GNU GP.

[20] Hojas de estilo en cascada (en inglés *Cascading Style Sheets*), CSS es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML.

accesibles. Hoy en día casi cualquier proveedor de alojamiento *(hosting)* dispone de las características mínimas requeridas por Drupal para su instalación y correcto funcionamiento. (Rodríguez, 2012)

A continuación se enuncian algunas de las características que posee este CMS. (Reyero, 2009)

Características generales:

- **Ayuda on-line:** Presenta un robusto sistema de ayuda en línea y páginas de ayuda para los módulos del núcleo, tanto para usuarios como para administradores.
- **Búsqueda:** Todo el contenido en Drupal es totalmente indexado en tiempo real y se puede consultar en cualquier momento.
- **Módulos:** La comunidad de Drupal ha contribuido con infinidad de módulos que proporcionan funcionalidades como página de categorías, autenticación mediante chat, mensajes privados, *bookmarks*, entre otros.
- **Personalización:** Presenta un robusto entorno de personalización que se encuentra implementado en el núcleo de Drupal, tanto el contenido como la presentación pueden ser individualizados de acuerdo con las preferencias definidas por el usuario.

Gestión de usuario:

- **Autenticación de usuarios:** Los usuarios se pueden registrar e iniciar sesión de forma local o utilizando un sistema de autenticación externo como *Jabber, Blogger, LiveJournal* u otro sitio de Drupal. Para su uso en una intranet, Drupal se puede integrar con un servidor LDAP[21].
- **Permisos basados en roles:** Los administradores de Drupal no tienen que establecer permisos para cada usuario. En lugar de eso, pueden asignar permisos a un rol y agrupar los usuarios por roles.

Gestión de contenidos:

- **Control de versiones:** El sistema de control de versiones de Drupal permite seguir y auditar totalmente las sucesivas actualizaciones del contenido: qué se ha cambiado, la hora y la fecha, quién lo ha cambiado, y más. También permite mantener comentarios sobre los sucesivos cambios o deshacer los cambios recuperando una versión anterior.
- **Enlaces permanentes (*Permalinks*):** Todo el contenido creado en Drupal tiene un enlace permanente asociado a él para que pueda ser enlazado externamente sin temor de que el enlace falle en el futuro.

[21] Protocolo Ligero de Acceso a Directorios.

- **Objetos de contenido (Nodos):** El contenido creado en Drupal es funcionalmente un objeto (nodo). Esto permite un tratamiento uniforme de la información como una misma cola de la página principal o permitir comentarios o no sobre cada objeto.
- **Plantillas (*Templates*):** El sistema de temas de Drupal separa el contenido de la presentación permitiendo controlar o cambiar fácilmente el aspecto del sitio web. Se pueden crear plantillas con HTML y/o con PHP.
- **Sindicación del contenido:** Drupal exporta el contenido en formato RDF/RSS para ser utilizado por otros sitios web. Esto permite que cualquiera con un 'Agregador de Noticias', pueda visualizar el contenido publicado en la web desde el escritorio.

Plataforma:

- **Multiplataforma:** Drupal ha sido diseñado desde el principio para ser multiplataforma. Puede funcionar con Apache[22] o Microsoft IIS [23]como servidor web y en sistemas Linux, BSD[24], Solaris, Windows y Mac OS X. Por otro lado, al estar implementado en PHP, es totalmente portable.
- **Independencia de la base de datos:** Aunque la mayor parte de las instalaciones de Drupal utilizan MySQL, existen otras opciones. Drupal incorpora una capa de abstracción de base de datos que actualmente está implementada y mantenida para MySQL y PostgreSQL[25], aunque permite incorporar fácilmente soporte para otras bases de datos.

Rendimiento y escalabilidad:

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

1.5.2 CMS Joomla

Joomla es uno de los más serios competidores de Drupal. Sus orígenes son anteriores a Drupal (fue creado a partir de Mambo), lo que le permitió posicionarse antes en el mercado. (Rodríguez, 2012)

[22] Servidor web HTTP de código abierto para plataformas *Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh* y otras.
[23] (*Internet Information Services* por sus siglas en inglés*)* son servicios para los ordenadores que funcionan con Windows.
[24] *Berkeley Software Distribution* (BSD*)*. En español, Distribución de Software *Berkeley*.
[25] Sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD.

Es un Sistema Gestión de Contenidos Avanzado, con las características y las funcionalidades que se encuentran en la mayoría de las aplicaciones de gama alta, a lo que se añade la flexibilidad y facilidad de uso.

Los administradores no necesitan conocimientos técnicos sobre la Tecnología de Internet (TI), para gestionar un sitio Joomla. Los editores de contenidos no necesitan conocimientos sobre la edición web, como el código HTML, para publicar textos completamente formateados. La gestión de Joomla se basa en navegadores web, y por tanto las actualizaciones y modificaciones del sitio pueden realizarse desde cualquier conexión a Internet disponible. (Centro de Documentación de Joomla, 2014)

Funciones Generales:

- Gestión del contenido del sitio basado completamente en una base de datos.
- Todas las secciones de noticias, productos o servicios se pueden editar y gestionar.
- Las secciones de temas pueden ampliarse mediante aportaciones de autores.
- Administre los usuarios con varios tipos de cuenta de usuario disponibles.
- Las características de etiquetado de los contenidos permiten un acceso flexible para cada tipo de usuario.
- Diseños completamente personalizables, incluyendo los menús izquierdo, derecho y central.
- Permite añadir imágenes a la galería del servidor vía navegador para su uso en cualquier lugar del sitio.
- Permite realizar una búsqueda con texto completo a través de todas las áreas de contenido.
- Espacios dinámicos de Foros/Encuestas/Votaciones.
- Funciona en Linux, FreeBSD, servidor MacOSX, Solaris y AIX[26].

1.5.3 Selección de CMS

La elección de uno u otro CMS no es sencilla. Ambos tienen una curva de aprendizaje elevada, especialmente para desarrolladores que quieren profundizar en su arquitectura y programar funcionalidades específicas. Sin embargo, Drupal ofrece una interfaz más amigable para los usuarios finales. La capacidad de crear sitios web fácilmente gestionables por los usuarios finales, que no tienen por qué tener conocimientos avanzados en informática y programación, fue uno de los factores de diferenciación que hizo inclinar la balanza hacia Drupal. Además, es quizás el CMS con la comunidad

[26] Es un sistema operativo UNIX abierto que permite ejecutar las aplicaciones deseadas en cualquier hardware y servidores IBM UNIX.

de usuarios y desarrolladores más activa que incorpora nuevas versiones y mejoras constantemente. Las nuevas versiones de este CMS ha evolucionado de tal forma que, además de ser capaz de realizar todas las funciones propias de un Sistema de Gestión de Contenido, permite la integración de muchas otras funciones avanzadas, más enfocadas a la interacción entre los usuarios, aspecto tenido en cuenta para su elección.

1.6 Lenguajes de desarrollo.

Teniendo en cuenta las características del sistema que se desea implementar, que tiene que ser un sistema Web, se ha decidido adoptar una arquitectura cliente-servidor.

Una arquitectura cliente-servidor se caracteriza por: El cliente y el servidor pueden actuar como una sola entidad y también como entidades separadas, realizando actividades o tareas independientes. El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa. El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo. Los cambios en el servidor implican pocos o ningún cambio en el cliente. (Becerra, y otros, 2013)

Un lenguaje de programación es una técnica estándar de comunicación que permite expresar las instrucciones que han de ser ejecutadas en una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen un lenguaje informático.

Hoy en día existen un gran número de lenguajes de programación que permiten desarrollar una aplicación con las características planteadas anteriormente, estos pueden dividirse en dos grandes grupos: lenguajes del lado del cliente y lenguajes del lado del servidor, dentro de este último grupo pueden mencionarse: PHP, ASP[27], Java, entre otros.

1.6.1 Lenguajes del lado del cliente

HTML: El Lenguaje de Etiquetado de Hipertexto es un estándar reconocido en todo el mundo y cuyas normas define un organismo sin ánimo de lucro llamado W3C[28]. Como se trata de un estándar reconocido por todas las empresas relacionadas con el mundo de Internet, una misma página HTML se visualiza de la misma manera en cualquier navegador de cualquier sistema operativo. La W3C define el lenguaje HTML como *"Un lenguaje comúnmente utilizado para la publicación de hipertexto en la Web y desarrollado con la idea de que cualquier persona o tipo de dispositivo pueda acceder a la información*

[27] (*Active Server Pages*) es la tecnología para la creación de páginas dinámicas del lado del servidor desarrollada por Microsoft.

28 Del inglés *World Wide Web Consortium.*

en la Web. HTML utiliza etiquetas que marcan elementos y estructuran el texto de un documento." (W3C, 2014)

La última versión de este lenguaje es la conocida HTML5, la misma está pensada con una mayor integración con los lenguajes CSS y JavaScript. HTML5 ha revolucionado la web y no solo se ve como el presente, sino como el futuro, por las numerosas **novedades** que trae con respecto a la versión anterior, entre las que se encuentran:

- Incluye nuevas etiquetas que permiten representar elementos familiares de las páginas, tal es el caso de: <*header*> para el encabezado de las páginas, <*nav*> para la navegación, <*section*> para una sección de la página y <*figure*> para asignar un título a una imagen.
- Incluye etiquetas para incorporar contenido multimedia como <*audio*> para audio y <*video*> para video, las cuales permiten la reproducción por parte del navegador de este tipo de contenido.
- Incorpora nuevas APIs[29] que facilitan en gran medida el trabajo de los desarrolladores, entre las que destacan la Geolocalización, *Drag and Drop* y el trabajo con Bases de Datos locales.

HTML5 propone estándares para cada aspecto de la web y también un propósito claro para cada una de las tecnologías involucradas. Con HTML5, HTML provee los elementos estructurales, CSS se encarga de volver esa estructura utilizable y atractiva a la vista, y JavaScript tendrá el poder necesario para proveer dinamismo y construir aplicaciones web completamente funcionales y con mayor capacidad de interacción con el usuario. (Musciano, 1999)

La selección de HTML5 como el lenguaje para realizar el maquetado de la propuesta de solución está sustentada en las novedosas herramientas que incorpora para el desarrollo web y el hecho de ser este un estándar establecido por la W3C.

JavaScript: Es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java. Legalmente, JavaScript es una marca registrada de la empresa Sun Microsystems. (Pérez, 2009)

29 Del inglés Application Programming Interface.

CSS[30]: Es un lenguaje de hojas de estilo creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML[31]. CSS3 es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas. Separar la definición de los contenidos y la definición de su aspecto presenta numerosas ventajas, ya que obliga a crear documentos HTML/XHTML bien definidos y con significado completo (también llamados "documentos semánticos"). Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes. (Pérez, 2008)

Librería JQuery: es una biblioteca o marco de trabajo de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM[32], manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Fue presentada el 14 de enero de 2006 en el *BarCamp* NYC.

Es software libre y de código abierto, posee un doble licenciamiento bajo la Licencia MIT[33] y la Licencia Pública General de GNUv2, permitiendo su uso en proyectos libres y privativos. JQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en Java Script que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio, la versión seleccionada será la 2.3. (Murphey, 2010)

Consiste en un único fichero JavaScript que contiene las funcionalidades comunes de DOM, eventos, efectos y AJAX. La característica principal de la biblioteca es que permite cambiar el contenido de una página web sin necesidad de recargarla, mediante la manipulación del árbol DOM y peticiones AJAX. JQuery implementa una serie de clases que permiten programar sin tener en cuenta el navegador con que trabaje el usuario, ya que funciona de la misma forma en todas las plataformas.

[30] Hojas de Estilo en Cascada (*Cascading Style Sheets*, por sus siglas en español)
[31] Siglas del inglés *eXtensible HyperText Markup Language*. XHTML es básicamente HTML expresado como XML válido
32 *Document Object Model* o DOM ('Modelo de Objetos del Documento' o 'Modelo en Objetos para la Representación de Documentos') es esencialmente una interfaz de programación de aplicaciones (API) que proporciona un conjunto estándar de objetos para representar documentos HTML y XML
33 Es una de tantas licencias de software que ha empleado el Instituto Tecnológico de Massachusetts (*MIT, Massachusetts Institute of Technology*)

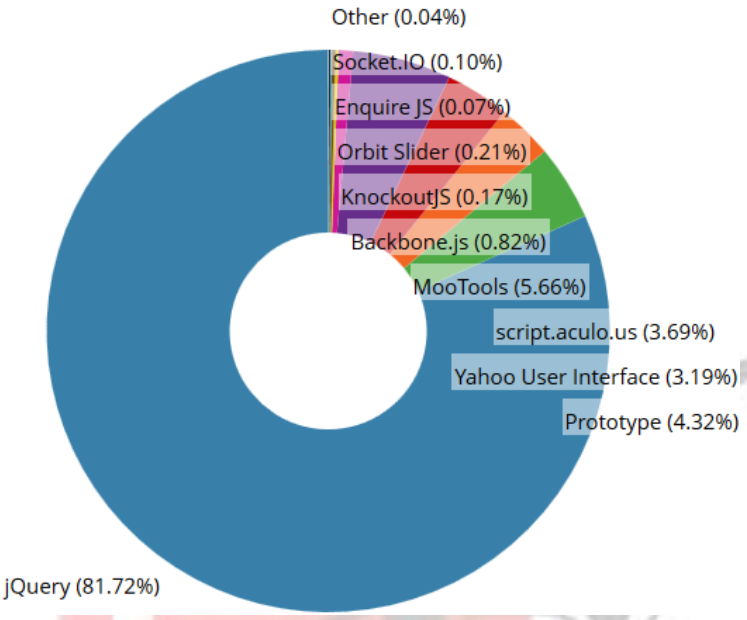

Ilustración 4. Estadísticas del uso de librerias JavaScript.

AJAX: Ajax es una nueva tecnología que es nombrada por primera vez por Jesse James Garret, es el acrónimo para *Asynchronous JavaScript + XML*, que en realidad no es una tecnología sino la combinación de muchas tecnologías como son: HTML, XHTML y CSS para la presentación de la información, el DOM y JavaScript que permiten el intercambio directo con la información y *XMLHttpRequest* que es el protocolo sobre el que está apoyado, y muy importante, es verdaderamente el corazón de Ajax. (Eguiluz, 2014)

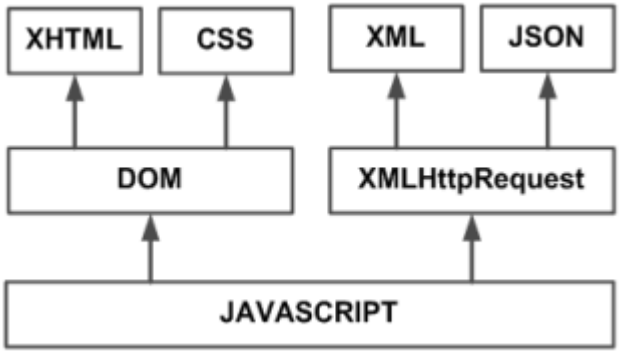

Ilustración 5. Diagrama de las tecnologías que componen AJAX.

Desarrollar aplicaciones AJAX requiere un conocimiento avanzado de todas y cada una de las tecnologías anteriores. En las aplicaciones web tradicionales, las acciones del usuario en la página (pinchar en un botón, seleccionar un valor de una lista, etc.) desencadenan llamadas al servidor. Una vez procesada la petición del usuario, el servidor devuelve una nueva página HTML al navegador del usuario.

AJAX permite mejorar completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página, ya que el intercambio de información con el servidor se produce en un segundo plano.

Las aplicaciones construidas con AJAX eliminan la recarga constante de páginas mediante la creación de un elemento intermedio entre el usuario y el servidor. La nueva capa intermedia de AJAX mejora la respuesta de la aplicación, ya que el usuario nunca se encuentra con una ventana del navegador vacía esperando la respuesta del servidor. (Eguiluz, 2014)

1.6.2 Lenguaje del lado del servidor:

PHP: Es un lenguaje de código abierto interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor. PHP5 puede hacer cualquier tarea que se pueda realizar con un script CGI[34], como procesar la información de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies. (Group, 2014)

Características importantes:

- Soporte para una gran cantidad de bases de datos: MySQL, PostgreSQL, Oracle[35], MS SQL Server, SybasemSQL[36], Informix, entre otras.
- Integración con varias bibliotecas externas, permite generar documentos en PDF (documentos de Acrobat Reader) hasta analizar código XML.
- Ofrece una solución simple y universal para las paginaciones dinámicas del Web de fácil programación.
- Perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes.
- Soportado por una gran comunidad de desarrolladores, como producto de código abierto, PHP goza de la ayuda de un gran grupo de programadores que permite que los fallos de funcionamiento se encuentren y reparen rápidamente.
- El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.

34 Del inglés *Common Gateway Interface*.
35 Fabricado por *Oracle Corporation*, utiliza la arquitectura cliente/servidor y ha incorporado en su sistema el modelo objeto-relacional.
36 Software de base de datos relacional fabricado y vendido por Sybase Inc

Con PHP se puede hacer cualquier cosa que se pueda realizar con un script CGI[37], como el procesamiento de información en formularios, foros de discusión, manipulación de cookies y páginas dinámicas.

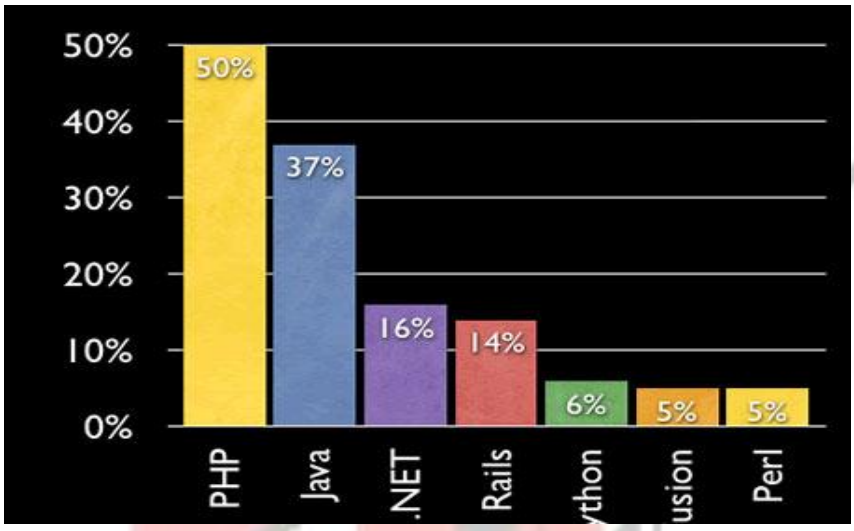

Ilustración 6. Funcionamiento de PHP.

1.7 Sistema Gestor de Base de Datos (SGBD).

La información de Drupal depende de la base de datos, cada información se encuentra en una tabla dentro de la base de datos. Por ejemplo, la información básica de los nodos se encuentra en la tabla de *Node*.

Desde sus primeras versiones Drupal soporta bases de datos MySQL y PostgreSQL, siendo ambos gestores de bases de datos los más usados para almacenar la información gestionada por un sistema desarrollado sobre Drupal.

1.7.1 PostgreSQL

Es un sistema de gestión de bases de datos (SGBD) objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones ha demostrado sus potencialidades frente a otras bases de datos comerciales.

[37] Es un programa que se ejecuta en un servidor web, accionado por la entrada procedente de un navegador.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. (Guerrero, 2010)

Ventajas de PostgreSQL

PostgreSQL ofrece muchas ventajas respecto a otros sistemas de bases de datos, entre las cuales se pueden encontrar:

- Instalación ilimitada: PostgreSQL no tiene costos asociados a la licencia del software, lo cual posibilita que se pueda instalar en varios servidores sin violar ningún acuerdo de licencia.
- Soporte técnico: Además de numerosas ofertas de soporte por parte de empresas de software libre, existen importantes comunidades de profesionales y entusiastas de PostgreSQL de las que se pueden obtener beneficios y contribución.
- Ahorros considerables en costos de operación: El software es diseñado y creado para tener presente un mantenimiento y ajuste mucho menor que los productos de los proveedores comerciales, conservando todas las características, estabilidad y rendimiento.
- Estabilidad y confiabilidad: En contraste a muchos sistemas de bases de datos comerciales es extremadamente común que las compañías reporten que PostgreSQL no presenta caídas en varios años de operación de alta actividad.
- Extensible: El código fuente está disponible sin costo para todo aquel que necesite extender o personalizar PostgreSQL de alguna manera.
- Por su arquitectura de diseño, escala muy bien al aumentar el número de CPUs[38] y la cantidad de RAM[39].
- Soporta transacciones y desde la versión 7.0, claves ajenas (con comprobaciones de integridad referencial).
- Brinda soporte para *triggers* y procedimientos almacenados en el servidor.
- Soporta replicación de bases de datos asíncrona, realizando primero las transacciones en un "servidor maestro" para que se puedan actualizar en los "servidores esclavos" dando alta disponibilidad al sistema.
- Posee buen sistema de seguridad mediante la gestión de usuarios, grupos de usuarios, permisos y contraseñas.

[38] Unidad central de procesamiento, conocido por el Acrónimo en inglés de *central processing unit* (CPU), o simplemente el procesador o microprocesador.
[39] *Random Access Memory* (Memoria de Acceso Aleatorio), es donde el computador guarda los datos que está utilizando en el momento presente.

- Posee gran capacidad de almacenamiento.
- Existen herramientas o aplicaciones para gestionar o administrar el servidor y sus bases de datos con interfaces gráficas muy intuitivas o en modo de líneas de comandos.

1.7.2 MySQL.

Es un sistema de gestión de bases de datos relacional, fue creada por la empresa sueca MySQL AB, la cual tiene el copyright del código fuente del servidor SQL[40], así como también de la marca. MySQL es un software de código abierto, licenciado bajo la GPL de la GNU, aunque MySQL AB distribuye una versión comercial, en lo único que se diferencia de la versión libre, es en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de otra manera, se vulneraría la licencia GPL.

Inicialmente, MySQL carecía de algunos elementos esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de esto, atrajo a los desarrolladores de páginas web con contenido dinámico, debido a su simplicidad, de tal manera que los elementos faltantes fueron complementados por la vía de las aplicaciones que la utilizan. Poco a poco estos elementos faltantes, están siendo incorporados tanto por desarrolladores internos, como por desarrolladores de software libre. (MySQL, 2014)

En las últimas versiones se pueden destacar las siguientes características principales:

- El principal objetivo de MySQL es velocidad y robustez.
- Soporta gran cantidad de tipos de datos para las columnas.
- Gran portabilidad entre sistemas, puede trabajar en distintas plataformas y sistemas operativos.
- Cada base de datos cuenta con 3 archivos: Uno de estructura, uno de datos y uno de índice y soporta hasta 32 índices por tabla.
- Aprovecha la potencia de sistemas multiproceso, gracias a su implementación multihilo.
- Flexible sistema de contraseñas (*passwords*) y gestión de usuarios, con un muy buen nivel de seguridad en los datos.
- El servidor soporta mensajes de error en distintas lenguas.

1.7.3 Selección del Sistema Gestor de Bases de Datos.

Se decide emplear como SGBD PostgreSQL en su versión 9.2.4.1 basándose fundamentalmente en su escalabilidad y la estabilidad que alcanza este gestor de bases de datos mediante el uso de

[40] *Structured Query Language* (lenguaje de consulta estructurado), es un lenguaje surgido de un proyecto de investigación de IBM para el acceso a bases de datos relacionales.

multiprocesos. Además las carencias de MySQL con respecto al uso de transacciones, *trigger* y el manejo de integridad referencial hacen de PostgreSQL un mejor candidato.

1.8 Entorno de desarrollo.

Un IDE (entorno de desarrollo integrado) es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI. Los IDEs pueden ser aplicaciones por si solas o pueden ser parte de aplicaciones existentes.

1.8.1 NetBeans IDE.

NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios (¡y creciendo!) en todo el mundo. *Sun MicroSystems*[41] fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo su patrocinador principal.

NetBeans IDE es un entorno de desarrollo, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender sus funcionalidades. Es un producto de código abierto y gratuito para uso tanto comercial como no comercial. El código fuente está disponible para su reutilización de acuerdo con la licencia CDDL[42] v1.0 y la licencia GPLv2. (Oracle Corporation and/or its affiliates, 2013)

Entre sus características se encuentra un sistema de proyectos basado en control de versiones y refactorización. Todas las funciones del IDE son provistas por *plug-ins*, al igual que Eclipse.

A partir de la versión 6.0 soporta las tecnologías *Enterprise JavaBeans* (EJB) 3.0, JAX-WS 2.1, *Java Server Faces* 1.2, *Java Server Page* 2.1, JSP *Standard Tag Library* (JSTL) 1.1, el editor de código es más rápido y más inteligente. Ya a partir de la versión 6.8 NetBeans IDE brinda soporte para PHP 5.3.

1.8.2 Eclipse.

Eclipse comenzó como un proyecto de IBM Canadá. Fue desarrollado por OTI (*Object Technology International*) como reemplazo de *VisualAge* también desarrollado por OTI. En noviembre del 2001, se formó un consorcio para el desarrollo futuro de Eclipse como código abierto. En 2003, la fundación

[41] Empresa informática adquirida por *Oracle Corporation,* anteriormente parte de *Silicon Valley,* fabricante de semiconductores y software.
[42] *Common Development and Distribution License* de acuerdo a sus siglas en inglés.

independiente de IBM fue creada. Eclipse 3.0 (2003) seleccionó las especificaciones de la plataforma *Open Services Gateway Initiative* (OSGI) como la arquitectura de tiempo de ejecución.

En 2006 la fundación Eclipse coordinó sus 10 proyectos de código abierto, incluyendo la Plataforma 3.2, para que fueran liberados el mismo día. Esta liberación simultánea fue conocida como la liberación Callisto. La versión consecutiva a Callisto es Europa, que corresponde a la versión 3.3 de Eclipse.

Eclipse es un IDE de código abierto independiente de una plataforma, es una aplicación de cliente enriquecido, emplea *plug-ins* para proporcionar toda su funcionalidad, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Eclipse es, en el fondo, únicamente un armazón sobre la que se pueden montar herramientas de desarrollo. La arquitectura de *plug-ins* permite, además de integrar diversos lenguajes, introducir otras aplicaciones accesorias que pueden resultar útiles durante el proceso de desarrollo, tales como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías, entre otros.

1.8.3 Selección del Entorno de Desarrollo.

Aunque ambos IDE son similares en cuanto a las funcionalidades y ventajas que ofrecen, se decidió escoger NetBeans en su versión 7.3 como IDE de desarrollo, debido a su ventaja respecto a Eclipse en cuanto a su integración con el lenguaje PHP, además de la experiencia adquirida por el equipo de trabajo en cuanto al uso de NetBeans IDE.

1.9 Servidor de aplicaciones. Apache.

Apache es el servidor web hecho por excelencia, su configurabilidad, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. (Apache, 2014) Este servidor ofrece varias características importantes a tener en cuenta, entre las cuales destaca: (Kair M., 2003)

- Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.
- Apache es una tecnología gratuita, *open source*. El hecho de ser gratuita es importante pero no tanto como que se trate de código fuente abierto. Esto le da una transparencia a este software de manera que si se quiere ver qué es lo que se instala como servidor, se puede saber, sin ningún secreto, sin ninguna puerta trasera.
- Apache es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades del servidor Web Apache. Actualmente existen muchos módulos para Apache que son adaptables a este, y están ahí para que se instalen cuando se necesiten. Otra cosa

importante es que cualquiera que posea una experiencia decente en la programación de C[43] o Perl [44]puede escribir un módulo para realizar una función determinada.

- Apache trabaja con gran cantidad de Perl, PHP y otros lenguajes de script. Perl destaca en el mundo del script y Apache utiliza su parte del pastel de Perl tanto con soporte CGI como con soporte *mod perl*. También trabaja con Java y páginas jsp. Teniendo todo el soporte que se necesita para tener páginas dinámicas.
- Apache te permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto.
- Tiene una alta configurabilidad en la creación y gestión de *logs*. Apache permite la creación de ficheros de *log* a medida del administrador, de este modo puedes tener un mayor control sobre lo que sucede en el servidor.

A pesar de las grandes ventajas que proporciona *Internet Information Server* (IIS) en cuanto a seguridad y servicios web; es privativo, lo que ha llevado a la directiva del proyecto SIGEF a utilizar como servidor web el Apache en su versión 9.1, que además de ser seguro, es uno de los más utilizados en el mundo puesto que es tan potente como flexible, es libre y de código abierto y además posee módulos configurables de forma tal que se pueda decidir cuáles de estos serán ejecutados en el servidor.

1.10 Conclusiones.

En el presente capítulo se realizó un estudio de sistemas similares, lo cual permitió identificar características necesarias a tener en cuenta para la propuesta de solución. Se arribó a la conclusión de que el proceso de desarrollo del *software* a construir será guiado por la metodología XP, atendiendo fundamentalmente a las características del equipo de desarrollo, el tiempo de desarrollo y la ocurrencia de cambios. El uso de Drupal 7 como CMS ofrecerá más que probadas capacidades para la creación, desarrollo y mantenimiento de una herramienta de gestión de este tipo, posibilitando fundamentalmente disminuir el tiempo de desarrollo. Se seleccionó HTML5, JavaScript y CSS3 como lenguajes del lado del cliente, además del marco de trabajo JQuery en su versión 2.3 y la tecnología AJAX, y del lado del servidor el lenguaje PHP5, contando así con el soporte necesario para la construcción de la propuesta de solución. La elección de PostgreSQL en su versión 9.2.4.1 como SGBD posibilitará gran estabilidad mediante el uso de multiprocesos, y permitirá la seguridad y persistencia de la información que

[43] También conocido como "Lenguaje de programación de sistemas" desarrollado en el año 1972 por Dennis Ritchie para UNIX.

[44] Perl *(Practical Extraction and Report Language)* es un lenguaje de programación desarrollado a finales de los años 80 por Larry Wall a partir otras herramientas de UNIX para la administración de tareas propias de sistemas.

gestionará el sistema a desarrollar. El empleo del entorno de desarrollo integrado Netbeans en su versión 7.3, posibilitará buena integración con las tecnologías seleccionadas y permitirá la codificación de la propuesta de solución. La selección de Apache en su versión 9.1 como servidor de aplicaciones web, permitirá desplegar el sistema sobre cualquier sistema operativo y proveerá la disponibilidad de los servicios del sistema una vez desarrollado.

Capítulo 2: Fundamentación Teórica: Exploración, Planificación e Iteraciones

2.1 Descripción de la propuesta de solución.

La propuesta de solución tiene como objetivo mejorar la calidad del proceso de gestión de los eventos investigativos en la Universidad. Dentro del sistema los usuarios podrán desempeñar los siguientes roles: Administrador, Miembro de la Comisión, usuario autenticado, usuario anónimo. Inicialmente el acceso se realizará mediante la autenticación de un usuario.

El usuario con rol Administrador será el encargado de crear los eventos, crear y asignar comisiones, gestionar la información de los eventos, gestionar todo el sistema de notificaciones. Además tiene la responsabilidad de asignar los permisos de acuerdo a los roles definidos previamente en el sistema. El usuario con rol Miembro de la comisión tiene a su cargo todo el proceso de revisión y calificación de un trabajo además de ser los encargados de decidir si un trabajo califica para concursar en un evento. La consulta de información almacenada en el sistema será permitido para todos los roles definidos, además de la posibilidad de participación en un evento, solo restringido esto último para los usuarios anónimos y para los miembros de la comisión del evento en cuestión.

La herramienta tendrá implementado 5 módulos: Calendario, Comisión, Entity_example, Trabajos_premiados, Trabajo_evento y Temáticas, además de usar 13 módulos de DRUPAL, de los cuales se necesitará modificar 3 de ellos para lograr una mejor integración a la herramienta.

Módulos Modificados:

- *Entity_example_basic*: Se modificará para crear una entidad de tipo evento con los campos necesarios para el sistema.
- Smtp: Es el módulo que permitirá habilitar en la herramienta la opción de enviar notificaciones y se modificará para definir el dominio necesario.
- *Simplenew*: El módulo crea una publicación de un artículo o página y se modificará para que posibilite la creación de publicaciones de los eventos definidos en el módulo entity_example_basic.

2.2 Usuario del sistema.

Los usuarios del sistema son aquellas personas encargadas de interactuar con la aplicación con un objetivo específico. El sistema contendrá algunas restricciones específicas para cada uno de los usuarios. A continuación se describen todos los usuarios del sistema con sus responsabilidades.

Usuarios	Responsabilidades
Administrador	El administrador del sistema es el encargado de crear los eventos, crear y asignar comisiones, gestionar la información de los eventos, gestionar todo el sistema de notificaciones, Además tiene la responsabilidad de asignar los permisos de acuerdo a los roles definidos previamente en el sistema.
Miembro de la Comisión	Los miembros de la comisión son los encargados de todo el proceso de revisión y calificación de un trabajo además de ser los encargados de decidir si un trabajo califica para concursar en un evento. Es válido aclarar que todos los privilegios de los miembros de la comisión son solamente del evento al que pertenecen.
Usuario autenticado	Tendrá acceso a visualizar la información, además de poder participar en un evento. Solo tendrá permiso para modificar información de su perfil.
Usuario anónimo	Solo tendrá permiso para visualizar la información almacenada sobre un tema en particular, pero no podrá participar en ningún evento.

***Tabla#2. Usuarios del sistema*.**

2.3 Diagrama conceptual del negocio.

Un diagrama conceptual del negocio no es más que un artefacto construido bajo las reglas de UML durante la concepción de un proyecto informático. Este modelo puede ser utilizado para capturar y expresar el entendimiento ganado sobre el negocio en cuestión.

El equipo de desarrollo consideró necesario generar este artefacto para un mejor entendimiento de la solución a desarrollar, a partir de identificar los conceptos y objetos relacionados con la gestión de eventos investigativos en la UCI.

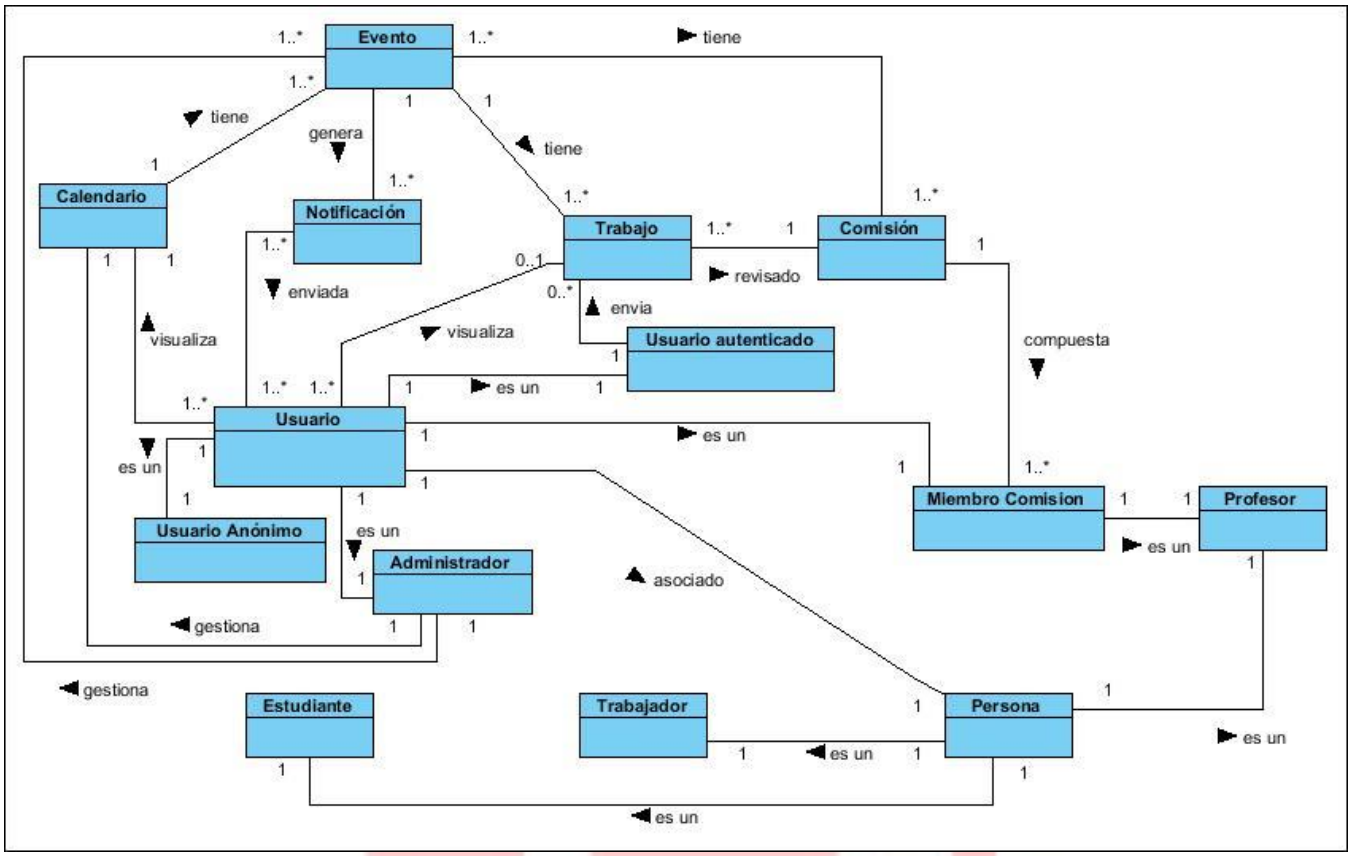

Ilustración 7. Modelo conceptual.

2.4 Historia de usuario.

XP utiliza para representar los requisitos del sistema un artefacto denominado Historias de Usuario (en adelante HU). Estas no son más que tarjetas de papel en las cuales el cliente describe las características del sistema. En cualquier momento del desarrollo las HU pueden ser modificadas o reemplazadas por otras más generales o específicas, o incluso añadir nuevas. Cada una es lo suficientemente entendible y está delimitada como para que los desarrolladores puedan implementarlas en unas semanas. (Jeffries, 2001) No lograr determinar todas las HU desde un inicio no es un problema, puesto que al inicio de cada iteración se registran los cambios en estas y a partir de ahí se planifica la próxima iteración. Las historias se descomponen en tareas de programación que son asignadas a los programadores para implementarlas durante una iteración.

2.4.1 Plantillas de historia de usuario

La metodología XP no define una plantilla específica para representar las HU, por lo que se definió una plantilla que contendrá, el nombre de la HU, el número, el usuario que realiza la acción, la estimación del tiempo para realizar la actividad, la prioridad, el riesgo, la iteración en que se encuentra, una breve descripción de la HU y un apartado para agregar alguna observación en caso que fuera necesario.

Historia de usuario	
Número: 1	**Nombre:** Gestionar eventos
Usuario: Administrador	
Prioridad: Alta	**Riesgo:** Alto
Estimación: 5 días	**Iteración:** 1
Descripción: El sistema permitirá crear evento, ver evento, modificar evento y eliminar evento, esto solo lo puede hacer el administrador. Una vez creado el evento el usuario solo podrá verlo.	
Observación:	

Tabla 3. HU-Gestionar Evento.

Historia de usuario	
Número: 2	**Nombre:** Gestionar Notificación
Usuario: Administrador	
Prioridad: Alta	**Riesgo:** Alto
Estimación: 3 días	**Iteración:** 2
Descripción: El sistema genera una notificación vía correo electrónico donde el usuario puede ver los estados de un trabajo en concurso. Esto puede ser gestionado solo por el administrador.	
Observación: La notificación será enviada a los usuarios registrados en el sistema.	

Tabla 4. HU-Gestionar Notificación.

Historia de usuario	
Número: 3	**Nombre:** Gestionar Comisión de eventos
Usuario: Administrador	
Prioridad: Alta	**Riesgo:** Alto
Estimación: 5 días	**Iteración:** 3
Descripción: El sistema tendrá las opciones de crear comisiones de evento, ver comisiones de evento, modificar comisiones de evento y eliminar comisiones de evento. Esta funcionalidad solo podrá ser realizada por el administrador.	
Observación:	

Tabla 5. HU-Gestionar Comisión de Eventos.

Historia de usuario	
Número: 4	**Nombre:** Gestionar usuario

Usuario: Usuario	
Prioridad: Alta	**Riesgo:** Alto
Estimación: 3 días	**Iteración:** 2
Descripción: El sistema permitirá que el usuario pueda ver el perfil de usuario, modificar el perfil de usuario y eliminar el perfil de usuario. El sistema tiene definido los roles y las funcionalidades específicas para cada usuario	
Observación: El proceso de crear usuario se hará mediante el registro en el sistema por parte del usuario.	

Tabla 6. HU-Gestionar roles y usuarios.

Historia de usuario	
Número: 5	**Nombre:** Gestionar Calendario
Usuario: Administrador	
Prioridad: Alta	**Riesgo:** Alto
Estimación: 3 días	**Iteración:** 1
Descripción: El sistema mostrara un calendario con los eventos programados en el mes en curso donde el usuario podrá consultar la información de cada evento en el calendario. Solo podrá ser gestionado por el administrador.	
Observación:	

Tabla 7. HU-Gestionar calendario.

Historia de usuario	
Número: 6	**Nombre:** Gestionar Proceso de revisión, aprobación y clasificación de trabajo
Usuario: Administrador	
Prioridad: Alta	**Riesgo:** Alto
Estimación: 5 días	**Iteración:** 3
Descripción: El sistema permitirá realizar un seguimiento detallado del estado de los trabajos asociados a eventos en curso. Además dará la posibilidad de realizar cambios de estados para un trabajo. Los cambios de estados de un trabajo los realiza un tribunal.	
Observación	

Tabla 8. HU-Gestionar Proceso de revisión, aprobación y clasificación de trabajos.

Historia de usuario	
Número: 7	**Nombre:** Gestionar Búsqueda por metadatos
Usuario: Usuario	
Prioridad: Alta	**Riesgo:** Alto
Estimación: 5 días	**Iteración:** 1
Descripción: El sistema permitirá que el usuario pueda realizar búsquedas filtradas por temáticas ya definidas, o por otros datos como autor, títulos, eventos, etc.	
Observación:	

Tabla 9. HU-Gestionar búsqueda por metadatos.

Historia de usuario	
Número: 8	**Nombre:** Autenticar usuario
Usuario: Usuario	
Prioridad: Alta	**Riesgo:** Alto
Estimación: 3 días	**Iteración:** 2
Descripción: El sistema permitirá al usuario autenticarse mediante un nombre de usuario único y una contraseña, como única vía para poder interactuar con el contenido del sistema que necesite de previa autenticación. El sistema cargará automáticamente los permisos asociados al rol del usuario.	
Observación: Este proceso solo es válido para usuarios registrados en el sistema.	

Tabla 10. HU-Gestionar autenticar usuario.

Historia de usuario	
Número: 9	**Nombre:** Registrar usuario
Usuario: Usuario	
Prioridad: Alta	**Riesgo:** Alto
Estimación: 3 días	**Iteración:** 2
Descripción: El sistema permitirá a un usuario registrarse para posteriormente contar con los privilegios de acuerdo al rol que se le asigne en el sistema.	
Observación: Este proceso solo será necesario para los usuarios que acceden al sistema por primera vez	

Tabla 11. HU-Gestionar registrar usuario.

Historia de usuario	
Número: 10	**Nombre:** Gestionar información sobre trabajo.
Usuario: Usuario	
Prioridad: Alta	**Riesgo:** Alto
Estimación: 3 días	**Iteración:** 2
Descripción: El usuario podrá editar la información de su trabajo en caso de que haya existido un error en el proceso de introducir los datos.	
Observación: Este proceso solo será realizado por los usuarios a los trabajos de su autoría.	

Tabla 12. HU-Gestionar información sobre trabajo.

2.5 Planificación.

"Toda técnica de planificación de software debe tratar de crear visibilidad, de tal manera que todo individuo involucrado en el proyecto pueda realmente ver cuán largo es el proyecto. Esto significa que se necesitan metas claras, metas que no sean dudosas y que claramente representen progreso. Las metas deben además ser cosas que toda persona involucrada con el proyecto, incluyendo el cliente, pueda entender y aprender a confiar en éstas." (Beck, 2008)

En esta fase los desarrolladores y el cliente establecen los tiempos de implementación de cada HU, la prioridad con que serán desarrolladas y las historias que serán implementadas en cada versión del proyecto. Se realizará un plan de entregas que debe ser lo antes posibles, y con cada iteración, el cliente recibirá una nueva versión. De esta manera un error en la parte inicial del sistema puede solucionarse rápidamente.

2.5.1 Estimación de esfuerzo por historias de usuario

Para un buen desarrollo de la solución propuesta, se realizó una estimación por cada una de las historias de usuarios que se identificaron. Se tuvo en cuenta que un punto de estimación equivale a una semana de trabajo; es decir, de lunes a viernes 8 horas que equivale a 40 horas semanales. En la tabla # 13 se detalla el grado de esfuerzo por cada historia de usuario a desarrollar, mientras mayor es la puntuación, mayor será el esfuerzo por parte del equipo de desarrollo:

Historias de usuario	Puntos de estimación
HU1. Gestionar eventos.	3
HU2. Gestionar Notificación.	2
HU3. Gestionar Comisión de eventos.	2
HU4. Gestionar usuario.	1
HU5. Gestionar Calendario.	3
HU6. Gestionar Proceso de revisión, aprobación y clasificación de trabajo.	2
HU7. Gestionar Búsqueda por metadatos.	1
HU8. Autenticar usuario.	1
HU9. Registrar usuario.	1
HU10. Gestionar información sobre trabajo.	1

Tabla 13.Estimación de esfuerzo por historias de usuario

2.6 Iteraciones.

Todo proyecto que siga la metodología XP, se ha de dividir en iteración de aproximadamente 3 semanas de duración. Al comienzo de cada iteración se deben seleccionar las HU definidas en la planificación. También se deben seleccionar las HU que no pasaron el test de aceptación que se realizó al terminar la iteración anterior. Estas HU son divididas en tareas de entre 1 y 3 días de duración que se les asignarán a los programadores. En la primera iteración se seleccionan las HU que abarquen

los aspectos más importantes de la arquitectura global, es decir, las historias que hagan referencia a la construcción de la estructura de todo el sistema.(Pablo Calabria, 2003)

Está prohibido intentar adelantarse e implementar cualquier cosa que no esté planificada en la iteración en curso. Habrá suficiente tiempo para añadir la funcionalidad extra cuando sea realmente importante según el plan de entregas.

Iteraciones	Orden de las HU a implementar	Cantidad de tiempo de trabajo
Iteración 1	HU1.Gestionar evento HU5.Gestionar calendario HU7.Gestionar búsqueda por metadatos	3 semanas
Iteración 2	HU8.Autenticar usuario HU9.Registrar usuario HU2.Gestionar notificación HU4.Gestionar usuario	3semanas
Iteración 3	HU3.Gestionar comisión de evento HU6.Gestionar Proceso de revisión, aprobación y clasificación de trabajo HU10. Gestionar información de trabajos	3 semanas

Tabla 14. Plan de iteraciones.

2.7 Plan de Entregas.

El cliente decide qué historias de usuario colocar en la entrega y cuáles se implementarán después, mientras que los programadores se encargan de las estimaciones correspondientes a cada entrega.

El plan de entregas deber ser lo más honesto posible porque constituye un documento oficial por el cual los clientes le exigen a los desarrolladores la entrega de las distintas versiones del producto. Si un proyecto no es capaz de cumplir con el plan de entregas pactado podría perder muchísimo dinero, baja la moral del equipo de desarrollo y los clientes dejan de confiar, aumentando la incertidumbre y la desesperación de los clientes.

Entregable	Final 1ra iteración(1ra semana de febrero)	Final 2da iteración(1ra semana de marzo)	Final 3ra iteración(1ra semana de abril)	Final (Última semana de abril)
Sistema de Gestión de Eventos Investigativos	Versión 0.1	Versión 0.2	Versión 0.3	Versión 0.4

Tabla 15. Plan de Entregas.

2.8 Conclusiones.

Durante el desarrollo del presente capítulo se expusieron los artefactos generados durante las tres primeras fases de la metodología XP, asegurando una implementación por etapas correctamente descrita y detallada. La definición de los roles y permisos de los distintos usuarios del sistema, servirá de ayuda a la hora de codificar las funcionalidades asociadas con el acceso de los usuarios al sistema, así como permitir o denegar el acceso a ciertas funcionalidades. Se generó el Diagrama Conceptual del Negocio para un mejor entendimiento de la solución a desarrollar, se documentaron las HU necesarias para una mejor descripción de las características con que debe contar el sistema. Se establecieron las iteraciones para desarrollar las HU teniendo en cuenta el nivel de dificultad de cada una y la relación entre ellas y se realizó el plan de entrega de cada versión bajo las exigencias del cliente.

Capítulo 3: Implementación y pruebas.

3.1 Descripción de la Arquitectura.

Gran parte del éxito de Drupal deriva de su arquitectura modular que simplifica el desarrollo, la colaboración y la contribución de los usuarios. Este CMS se basa en la abstracción y organización en capas que aplica en el tratamiento de los contenidos para conseguir su reconocida flexibilidad y facilidad en la creación de sitios web. En lugar de considerar el sitio web como un conjunto de páginas interrelacionadas, se estructuran los contenidos en una serie de elementos básicos. (Drupal, 2014)

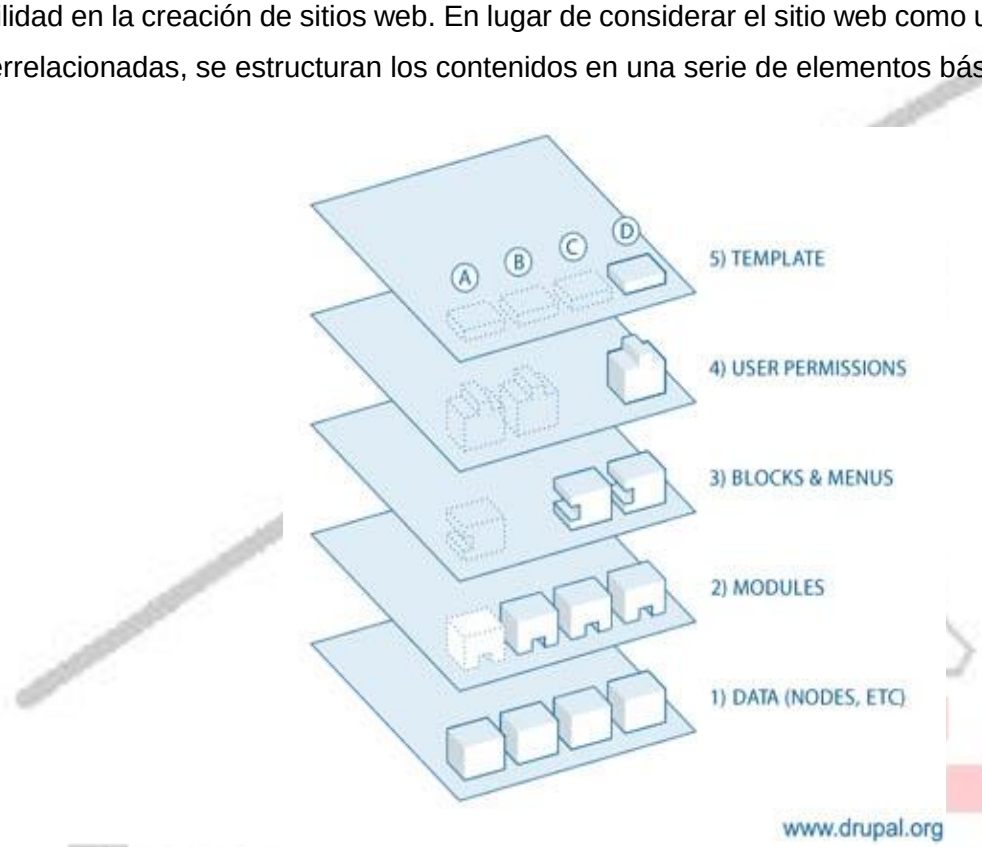

Ilustración 8.Arquitectura de Drupal.

Los Nodos (*Nodes*) son los elementos básicos en donde se almacena la información y los contenidos. Así a medida que el sitio web crece, lo va haciendo el número de nodos los cuales van formando un "depósito de *nodos*" cada vez mayor. Se puede decir que la primera capa de la estructura de CMS la forma este "deposito" de elementos.

Los Módulos (*Modules*) son los elementos que operan sobre los nodos y otorgan funcionalidad a Drupal permitiendo incrementar sus capacidades o adaptarlas a las necesidades de cada sitio web. Son como *Plug-Ins* que se instalan en el sitio web proporcionándole nuevas funcionalidades.

La siguiente capa la constituyen los Bloques y los Menús (*Blocks & Menus*). Estos permiten estructurar y organizar los contenidos en la página web. Es decir que son los elementos que albergan y permiten acceder al usuario a la salida generada y procesada por los módulos a partir de la información almacenada en los nodos.

La siguiente capa importante es la de control de usuarios y permisos. Actualmente, la mayor parte de sitios web son multiusuario, por lo que la seguridad y control de los usuarios es un punto clave para garantizar la integridad de la información almacenada. Con esta finalidad Drupal dispone de un registro de usuarios y de roles que permiten especificar que tareas pueden realizar y a que contenidos puede acceder cada tipo de usuario. Es decir que las operaciones que se pueden realizar sobre los elementos provenientes de las capas inferiores (lectura, modificación, creación…) se encuentran limitadas por la capa de control de usuarios y permisos de Drupal.

La última capa es la capa de Temas (*Themes*), es la que establece la apariencia gráfica o estilo de la información que se le muestra al usuario. Esta separación entre información y aspecto gráfico permite cambiar el diseño u apariencia del sitio web sin necesidad de modificar los contenidos, lo que es muy práctico si lo único que se necesita es renovar la apariencia de un sitio web.

Cron es otro elemento importante en Drupal. Es una aplicación que se ejecuta periódicamente con la finalidad de realizar las tareas básicas de mantenimiento del sitio web, como limpiar los *logs*, indexar los nuevos contenidos, etc. La configuración de Cron variará dependiendo del sistema operativo en que se esté trabajando, no obstante existen algunas alternativas, menos efectivas, pero más sencillas de utilizar Cron. Una de ellas es ejecutarlo manualmente seleccionando la opción "*Run cron Manually"* y otra opción es utilizar el "*Contributed module" "Poormanscron*". Este módulo hace que Cron se ejecute cada vez que el sitio recibe una visita, en lugar de hacerlo de forma periódica cada cierto tiempo. (Drupal, 2014)

3.2 Tarjetas Clase-Responsabilidades-Colaboradores (CRC).

Las tarjetas Clases, Responsabilidades y Colaboraciones (CRC) son una técnica de modelado orientado a objetos que permite identificar las clases, sus responsabilidades y colaboraciones, además ayudan al refinamiento de clases. Drupal no utiliza la programación orientada a objetos, en cuanto a la falta de clases explícitamente declaradas, pero en su diseño se reflejan muchos paradigmas de la POO como objetos, abstracción, encapsulamiento, polimorfismo y herencia. Sin embrago las tarjetas CRC también son muy útiles para mejorar la idea de la arquitectura de Drupal, permitiendo conocer las responsabilidades y la distribución de los módulos que serán desarrollados. Para adecuar las tarjetas a los módulo de este trabajo se adopta lo siguiente: las responsabilidades serán las funciones que realiza y sus colaboradores los métodos propios del módulo.

Responsabilidades	Colaboraciones
Módulo: Comisión	
Añadir comisión Añadir Integrante Eliminar comisión Eliminar integrante Listar comisión Listar integrantes	comision_entity_info ctools_ajax_load_comision comision_user_autocomplete comision_menu eliminar_integrante comision_form_crear comision_generate_integrantes comision_form_crear_validate comision_form_crear_submit comision_form_eliminar

Tabla 16. Tarjeta CRC#1.

Responsabilidades	Colaboraciones
Módulo: Entity_example	
Adicionar evento Eliminar evento Actualizar evento Listar Eventos	entity_example_entity_info entity_example_basic_load entity_example_basic_load_multiple entity_example_basic_uri entity_example_menu entity_example_info_page entity_example_permission entity_example_basic_list_entities entity_example_basic_title entity_example_basic_view entity_example_field_extra_fields entity_example_basic_add entity_example_basic_form entity_example_basic_form_validate entity_example_basic_form_submit entity_example_basic_save entity_example_basic_delete

Tabla 17. Tarjeta CRC#2.

Responsabilidades	Colaboraciones
Módulo: Trabajo_evento	
Adicionar trabajo Eliminar trabajo Actualizar trabajo Listar trabajo Asignar trabajo a comisión	trabajo_evento_entity_info trabajo_evento_menu trabajo_evento_form_crear trabajo_evento_form_crear_submit

Tabla 18. Tarjeta CRC#3.

Responsabilidades	Colaboraciones
Módulo: Trabajos_premiados	
listar trabajos premiar trabajos guardar trabajos premiados	trabajos_premiados_info trabajos_premiados_menu trabajos_premiados_form_crear obtener_eventos trabajos_premiados_form_crear_submit

Tabla 19. Tarjeta CRC#4.

Responsabilidades	Colaboraciones
Módulo: Temáticas	
Adicionar temáticas Eliminar temáticas	tematica_entity_info tematica_menu tematica_form_crear tematica_form_crear_submit

Tabla 20. Tarjeta CRC#5.

3.3 Diagrama Entidad Relación (DER).

Un diagrama de Entidad-Relación o Modelo de Datos es un conjunto de conceptos que sirven para describir la estructura de una base de datos, los datos las relaciones entre los datos y las restricciones que deben cumplirse sobre los datos.

Los modelos de datos aportan la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos. Un modelo de datos está orientado a representar los elementos que intervienen en la realidad o en un problema dado y en la forma que se relacionan estos elementos entre sí.

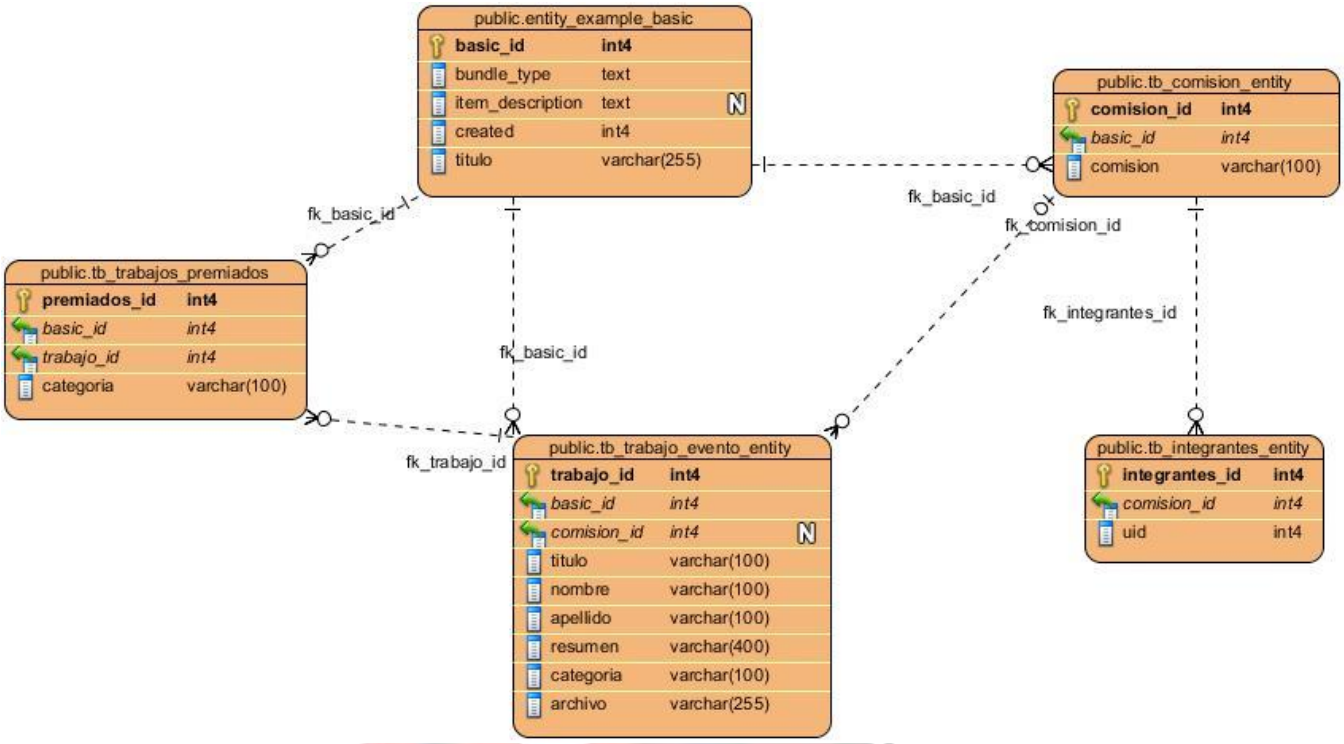

Ilustración 9.Entidad relación.

3.4 Patrones de Diseño.

Craig Larman define en la segunda edición de su libro "*UML y Patrones*" a un patrón como "*un par problema/solución con nombre que se puede aplicar en nuevos contextos, con consejos acerca de cómo aplicarlo en nuevas situaciones y discusiones sobre sus compromisos*." (Larman, 2003 pág. 162). Los patrones de diseño comunican los estilos y soluciones consideradas como "buenas prácticas", que los expertos en el diseño orientado a objetos utilizan para la creación de sistemas. (Larman, 2003).

3.4.1 Patrones GoF[45]

- ***Singleton***[46]: Es un patrón de tipo creación, ya que abstrae el proceso de creación de instancias. Resuelve el problema de que exista una instancia única de una clase, proporcionando un punto de acceso global a la misma. (Larman, 2003)

[45] Del inglés Gang of Four, en español Banda de los Cuatro. Es el nombre con el que se conoce comúnmente a los autores del libro Design Patterns (en español Patrones de Diseño) (ISBN 0-201-63361-2), los mencionados autores son: Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides.

[46] Conocido en español como "Instancia Única".

La esencia del patrón *Singleton* en la programación orientada a objetos consiste en tener una sola instancia de un objeto disponible para toda la aplicación que la contiene. Provee una instancia global permitiendo que otros objetos accedan a esta única instancia.

El patrón *Singleton* asegura que exista una única instancia de una clase (ver Anexo 5). A primera vista, se llega a pensar que pueden utilizarse clases con miembros estáticos para el mismo fin. Sin embargo, los resultados no son los mismos, ya que en este caso la responsabilidad de tener una única instancia recae en el cliente de la clase. El patrón *Singleton* hace que la clase sea responsable de su única instancia, quitando así este problema a los clientes.

Adicionalmente, si todos los métodos de esta clase son estáticos, éstos no pueden ser extendidos, desaprovechando así las capacidades polimórficas que proveen los entornos orientados a objetos.

El funcionamiento de este patrón es muy sencillo y podría reducirse a los **siguientes conceptos:**

- ✓ Ocultar el constructor de la clase *Singleton*, para que los clientes no puedan crear instancias.
- ✓ Declarar en la clase *Singleton* una variable miembro privada que contenga la referencia a la instancia única que se desea gestionar.
- ✓ Proveer en la clase *Singleton* una función o propiedad que brinde acceso a la única instancia gestionada por el *Singleton*. Los clientes acceden a la instancia a través de esta función o propiedad. (Welicki, 2014)

Ventajas: (Software, 2014)

- ✓ El acceso a la ¨ Instancia Única ¨está más controlado¨.
- ✓ Se reduce el espacio de nombres (frente al uso de variables globales).
- ✓ Permite refinamientos en las operaciones y la representación, mediante la especialización por herencia.
- ✓ Es fácilmente modificable para permitir más de una instancia, y en general, para controlar el número de las mismas.

Requisitos de usabilidad:

- ✓ Adecuado para cuando debe haber únicamente una instancia de la clase y debe ser claro su acceso para los clientes
- ✓ La ¨Instancia Unica ¨ debe ser especializado mediante herencia y los clientes deben poder usar la Instancia extendida sin tener que modificar su código (el de los clientes).

- ***Decorator***[47]: También conocido como *wrapper* o envoltorio es la alternativa a la herencia para decorar la responsabilidad de un subconjunto de objetos. Tiene como aplicabilidad añadir o eliminar funcionalidades a objetos de forma dinámica y transparente además de evitar una explosión de subclases para poder tener todas las combinaciones de una serie de funcionalidades independientes (ver Anexo 7).

  Ventajas:

  - ✓ Mayor flexibilidad que la herencia, incluso puede decorarse un objeto varias veces con un mismo decorador, lo que sería imposible con la herencia.
  - ✓ Favorece la definición de interfaces y clases bases ligeras, solo se carga con las funcionalidades que se necesitan, no son necesarias mega clases que sean capaces de hacerlo todo. (INFORMÁTICOS, 2014)

  Desventajas: (Software, 2014)

  - ✓ La Transparencia tiene el inconveniente, de que a pesar de ser diferentes por muchos motivos, un decorador y un componente son indistinguibles, por lo que no se puede confiar en la identidad de los objetos.
  - ✓ Gran fragmentación, el sistema se llena de gran cantidad de objetos pequeños, lo cual puede dificultar el aprendizaje de su funcionamiento y su depuración, aunque el que lo domine puede adaptarlo fácilmente.

  Requisitos de usabilidad:

  - ✓ Se quiere añadir responsabilidades a un objeto de manera dinámica y transparente (independientemente de otros objetos).
  - ✓ Cuando es imposible la extensión de funcionalidad por herencia, por ser aquella imprevisible en tipo y número.

- **Observer**[48]**:** EL patrón observador es uno de los más ampliamente utilizados en la programación de un sistema, aunque a veces, incluso el programador no sabe que lo está usando. Hay dos roles básicos en este patrón, el componente observable, que lanza eventos para notificar cambios en su estado, y el observador que espera y recibe estos eventos (ver Anexo 8). (Rodero, y otros, 2014)

  Un ejemplo muy frecuente en aquellos sistemas que se fragmentan en un conjunto de clases que cooperan es la necesidad de mantener la consistencia entre los distintos objetos

[47] Conocido en español como Decorador
[48] Conocido en español como Observador

interrelacionados. Para no recurrir a soluciones fuertemente acopladas (que reducen la posibilidad de reutilización), este patrón define una dependencia de uno-a-mucho entre objetos, para que, cuando uno de ellos cambie su estado, todos los que dependan de él sean avisados y puedan actualizarse convenientemente. (Software, 2014)

Este patrón funciona a través de un mediador llamado Gestor de Cambios (ver Anexo 6), que puede ser un gestor sencillo, que sea capaz de reenviar eventos desde cualquier objeto observable a todos los observadores. También puede ser más complejo, guardando información acerca de que evento interesan a que observadores. (Rodero, y otros, 2014)

Ventajas:

- ✓ Utiliza los ¨Métodos Plantilla¨ que es una técnica fundamental para la reutilización de código, especialmente en las librerías de clases, donde son la razón de ser de la ¨factorización de comportamiento común¨. Llevan a una estructura de control invertido, cuyo paradigma es el ¨Principio de *Hollywood*¨[49]. (Software, 2014)

Requisitos de usabilidad: (Software, 2014)

- ✓ Adecuado cuando se pretende implementar una sola vez las partes invariantes de un algoritmo, permitiendo a las subclases implementar el comportamiento que puede variar.
- ✓ Cuando se quiere ¨sacar factor común¨ del comportamiento compartido por varias clases para localizarlo en una única clase, para evitar la duplicación de código. A esta técnica se le conoce comúnmente como Refactorizar para Generalizar y sigue el esquema:
  - Identificar las diferencias de código ya existentes.
  - Separarlas, convirtiéndolas en operaciones nuevas.
  - Reemplazar las diferencias en el código existentes por un ¨Método Plantilla¨ qué invoque las operaciones que necesite.

3.5 Estilos y estándares de codificación.

Un estándar de codificación o estilo de programación es el conjunto de reglas o normas usadas para escribir código e incluye una gama de aspectos dentro del proceso de codificación. El uso de estilos de programación facilitará que los programas sean más fáciles de leer y comprender por otros

[49] Principio ¨No nos llame, ya le llamaremos¨, la clase padre invoca la implementación que la clase hija hace de operaciones primitivas. pero no al revés.

programadores, además, se reducirán el número de errores que se presentan durante el desarrollo del programa.

Las reglas de Drupal que se aplicarán al código dentro de Drupal y sus módulos de tercero, las cuales se basarán libremente en las normas de codificación de PHP *Extension and Application Repository* (PEAR).

Operadores:

- Todos los operadores binarios, que se encuentran entre valores como +, -, =, ==, >, deberán tener un espacio antes y después del operador, para facilitar la lectura. Por ejemplo, $foo = $bar.
- Los operadores unarios, que operan en un solo valor, tales como ++, no deberían tener un espacio entre el operador y el numero variable. Ejemplo, $a++.

Estructuras de control:

- Las sentencias de control deberán tener un espacio entre la palabra control y paréntesis de apertura para distinguirlas de las llamadas a funciones.
- Siempre se utilizarán las llaves, incluso en situaciones en las que son técnicamente opcionales. Aumentará la legibilidad y disminuirá la probabilidad de errores lógicos que se introducen cuando las líneas se añaden nuevos elementos.
- No se usará else if, siempre elseif.

```
If (condition1 || condition2) {
  action1,
}
elseif (condition3 && condition4) {
 action2,
}
else {
   defaultaction,
}
```

Longitud de la línea y el ajuste:

- En general, todas las líneas de código no deberán ser superiores a 80 caracteres.

Matrices o arreglos:

- Las matrices deberán tener el formato con un espacio separado cada elemento después de la coma, y los espacios alrededor del operador => clave de la asociación.
  $some_array = array('hello', 'world', 'foo' => 'bar');
- Se tendrá en cuenta que si en la línea se declara una matriz que se extiende por más de 80 caracteres, cada elemento debe ser dividido en su propia línea y sangría.
  $form['title'] = array(
  '#type' => 'textfield',
  '#title' => t('Title'),
  '#size' => 60,
  '#maxlength' => 128,
  '#description' => t('The title of your node.'),
  );
- Se tendrá en cuenta la coma al final del último elemento del *array*, no es un error. Esto ayudará a evitar errores de análisis, si otro elemento se coloca al final de la lista más adelante.

Comillas:

Drupal no tiene un duro estándar para el uso de comillas simples contra comillas dobles. Siempre que sea posible, mantener la coherencia dentro de cada módulo y respetar el estilo personal de otros desarrolladores. Con esta advertencia en mente, las cadenas de comillas simples son conocidas por ser más rápidas debido a que el analizador no tiene que buscar en líneas las variables. Su uso se recomienda excepto en dos casos:

- En la línea de uso variable, por ejemplo, "<h2> $header </ h2>".
- Cadenas traducidas, donde se podrá evitar el escape de comillas simple encerrando la cadena entre comillas dobles. Una cadena, sería "Él es una buena persona."

Concatenaciones de cadenas:

- Se utilizará siempre un espacio entre el punto y las partes concatenadas para mejorar la legibilidad.
  $string = 'Foo' . $bar;
  $string = $bar . 'foo';
  $string = 'foo' . 'bar';
- Al concatenar variables simples se podrá usar comillas dobles y añadir la variable dentro, de lo contrario se utiliza la comilla simple.
  $string = "Foo $bar";

- Cuando se utilice el operador de asignación de concatenación (.=), se procurará un espacio a cada lado como con el operador de asignación:
  $string. = 'Foo';
  $string. = $bar;

3.6 Tareas de Ingeniería.

XP propone dividir cada HU en tareas de ingeniería con el objetivo de facilitar la implementación a los programadores, estas tareas deben tener una duración de entre 1 y 3 días aproximadamente. Las tareas seleccionada por cada HU fueron (ver Anexo 4):

Tarea	
Número: 1	**Número de HU:** 1
Nombre: Adicionar evento.	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Fecha inicio: 12 de enero de 2014	**Fecha fin:** 13 de enero de 2014
Programador responsable: Edelso A. Rojas, Jorge M. Soldevila	
Descripción: Adicionar un evento al sistema.	

Tabla#21. Tarea# 1.

Tarea	
Número: 2	**Número de HU:** 1
Nombre: Ver evento.	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Fecha inicio: 14 de enero de 2014	**Fecha fin:** 14 de enero de 2014
Programador responsable: Edelso A. Rojas, Jorge M. Soldevila	
Descripción: Ver todos los evento del sistema.	

Tabla#22. Tarea# 2

Tarea	
Número: 3	**Número de HU:** 1
Nombre: Editar evento.	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Fecha inicio: 15 de enero de 2014	**Fecha fin:** 15 de enero de 2014
Programador responsable: Edelso A. Rojas, Jorge M. Soldevila	
Descripción: Editar los datos de los eventos.	

Tabla#23. Tarea# 3.

Tarea	
Número: 4	**Número de HU:** 1
Nombre: Eliminar evento	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Fecha inicio: 16 de enero de 2014	**Fecha fin:** 16 de enero de 2014
Programador responsable: Edelso A. Rojas, Jorge M. Soldevila	
Descripción: Eliminar el evento.	

Tabla#24. Tarea# 4.

3.7 Pruebas.

La metodología XP enfatiza en la realización de un sin números de pruebas a lo largo de todo el desarrollo del software, con el fin de lograr un producto con calidad, reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su corrección. En este proceso no solo participa el equipo de desarrollo, también es importante la colaboración del cliente, sobre todo en las pruebas de aceptación.

En XP las pruebas se dividen en dos grupos: pruebas unitarias encargadas de verificar el código y pruebas de aceptación que están orientadas a probar las funcionalidades del sistema. Un equipo de desarrollo que siga el proceso de XP debe primero probar su código de forma unitaria, integrar este a la aplicación y después realizar las pruebas de aceptación.

3.7.1 Desarrollo dirigido por pruebas (TDD)

El desarrollo dirigido por pruebas es una característica de la metodología XP para comprobar el funcionamiento de los códigos que se van implementando. Este estilo sigue el principio "*test-first*", significa que las pruebas se crean antes de comenzar la implementación. La idea es que, al pensar en cómo se probará la funcionalidad, se está pensando en la propia funcionalidad desde el punto de vista de su interfaz (qué métodos tendrá y con qué parámetros), ayudando a desarrollar así un mejor diseño. De ésta manera, además, se asegura de que no exista ninguna funcionalidad que no esté probada. Una vez creadas las pruebas el código no pasa a producción hasta que el resultado de las mismas no sea satisfactorio. Luego se inicia el proceso de refactorización que consiste en limpiar y organizar el código, adaptarlo a los patrones y aumentar su legibilidad, todo esto sin modificar su comportamiento externo. (Koskela, 2008)

3.7.2 Pruebas unitarias

Las pruebas unitarias clasificadas como pruebas de caja blanca dentro de la metodología de desarrollo XP, también llamadas pruebas modulares permiten determinar si un módulo del sistema está listo y

correctamente terminado. El objetivo fundamental de las pruebas unitarias es asegurar el correcto funcionamiento de las interfaces, o flujo de datos entre componentes teniéndose bien claro los siguientes aspectos a la hora de su aplicación:

- Los datos de entrada son conocidos por el Analista de Pruebas y estos deben ser preparados con minuciosidad, ya que el resultado de las pruebas dependen de estos.
- Se debe conocer que componentes interactúan en cada caso de prueba.
- Se debe conocer de antemano que resultados debe devolver el componente según los datos de entrada utilizados en la prueba.

Finalmente se deben comparar los datos obtenidos en la prueba con los datos esperados, si son idénticos el modulo supera la prueba.

3.7.2.1 Modulo Unit Testing.

Para la realización de las pruebas unitarias en el CMS Drupal, se utilizó el módulo *unit testing* el cual es una práctica que consiste en tomar una pequeña porción de código y someterlo a unas pruebas de programación para probar su corrección. Por lo general se someten a prueba, funciones o pequeñas funcionalidades específicas con un conjunto de parámetros controlados que servirán para hacer las comprobaciones necesarias en función de los valores de retorno de la funcionalidad testeada. Dentro de sus utilidades se pueden observar:

- Ayuda a producir un código de mayor calidad.
- Detección rápida de errores cuando se programan nuevas funcionalidades o se realizan cambios en el código.
- Sirve como pequeña fuente de documentación sobre qué es lo que se espera que haga el código.
- Realizar pruebas unitarias obliga al programador a escribir el código en pequeñas porciones con el fin de que puedan ser probadas independientemente.

3.7.3 Pruebas de aceptación

Las pruebas de aceptación son las especificaciones para el comportamiento deseado y la funcionalidad de un sistema. Muestran, por una HU dada, cómo el sistema se encarga de ciertas condiciones e insumos y con qué tipo de resultados. (KOSKELA, 2008). Los clientes junto a un

miembro del equipo de desarrollo son los responsables de verificar que los resultados de estas pruebas sean los correctos para así tomar decisiones acerca de las mismas. Una HU no se puede considerar terminada hasta que no pase los test de aceptación. Es recomendable publicar los resultados de las pruebas de aceptación, para que todo el equipo de desarrollo esté al tanto de esta información.

A continuación aparecen los casos de prueba de aceptación referente a la solución propuesta (ver Anexo 3):

Caso de prueba de aceptación	
Código: HU1_CP1	**Historia de usuario:** 1
Nombre: Adicionar evento	
Condiciones de ejecución: El usuario debe estar autenticado con el rol de Administrador del sistema.	
Entrada/ Pasos de ejecución: Entrada: La entrada consiste en los datos del evento que el usuario debe introducir para ser insertado en la base de datos. Pasos de ejecución: Para agregar un nuevo evento se debe seleccionar del menú principal la opción “Crear evento”, el sistema mostrara una interfaz con los campos de datos que se deben introducir, una vez introducido los datos se presiona el botón “Guardar” y seguido se mostrará un mensaje de confirmación además del evento añadido en una tabla.	
Resultado esperado: El evento es adicionado correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#25. Caso de prueba de aceptación#1.

Caso de prueba de aceptación	
Código: HU1_CP2	**Historia de usuario:** 1
Nombre: Ver evento	
Condiciones de ejecución: El usuario debe estar autenticado correctamente además de que debe existir algún evento adicionado en la Base de Datos.	
Entrada/ Pasos de ejecución: Pasos de ejecución: Para visualizar los datos de un evento se debe seleccionar del menú principal la opción “Administrar evento” y luego seleccionar el evento deseado de una tabla mostrada por el sistema con todos los eventos.	
Resultado esperado: Se mostrarán los datos del evento.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#26. Caso de prueba de aceptación#2.

Caso de prueba de aceptación

Código: HU1_CP3	**Historia de usuario:** 1
Nombre: Editar evento	
Condiciones de ejecución: El usuario debe estar autenticado con el rol de Administrador del Sistema además de que debe existir al menos un evento adicionado al sistema.	
Entrada/ Pasos de ejecución: Entrada: Los datos del evento que el usuario quiere modificar. Pasos de ejecución: Para visualizar los datos de un evento se debe seleccionar del menú principal la opción “Administrar evento” y luego seleccionar el evento deseado de una tabla mostrada por el sistema con todos los eventos y se modifican los campos necesarios, luego se presiona el botón “Guardar” y se mostrará un mensaje de confirmación además del evento modificado en una tabla.	
Resultado esperado: El evento es editado correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#27. Caso de prueba de aceptación#3.

Caso de prueba de aceptación	
Código: HU1_CP4	**Historia de usuario:** 1
Nombre: Eliminar evento	
Condiciones de ejecución: El usuario debe estar autenticado con el rol de Administrador del Sistema además de que debe existir al menos un evento adicionado al sistema.	
Entrada/ Pasos de ejecución: Pasos de ejecución: Para eliminar un evento se debe seleccionar del menú principal la opción “Administrar evento” y luego eliminar el evento deseado de la tabla.	
Resultado esperado: El evento es eliminado correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#28. Caso de prueba de aceptación#4.

Caso de prueba de aceptación	
Código: HU2_CP1	**Historia de usuario:** 2
Nombre: Enviar notificación.	
Condiciones de ejecución: El usuario debe estar registrado correctamente en el sistema y debe ocurrir al menos algún cambio de estado en el sistema.	
Entrada/ Pasos de ejecución	
Resultado esperado: La notificación es enviada correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#29. Caso de prueba de aceptación#5.

Caso de prueba de aceptación

Código: HU3_CP1	**Historia de usuario:** 3
Nombre: Adicionar Comisión de revisión	
Condiciones de ejecución: El usuario debe estar autenticado con el rol de Administrador del Sistema y además debe haber algún evento insertado en la base de datos.	
Entrada/ Pasos de ejecución: Entrada: Datos de los integrantes de la comisión. Pasos de ejecución: Seleccionar en el menú de navegación la opción “Comisiones”, se debe seleccionar el evento al que estará asociado la comisión a insertar, luego se presiona el botón “Adicionar comisión” y se desplegará un campo para entrar el nombre de la comisión y luego se presionar el botón “Añadir”, el sistema mostrará un mensaje de confirmación además de la nueva comisión en una tabla.	
Resultado esperado: La Comisión ha sido adicionada correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#30. Caso de prueba de aceptación#6.

Caso de prueba de aceptación	
Código: HU3_CP2	**Historia de usuario:** 3
Nombre: Ver comisión de revisión.	
Condiciones de ejecución: El usuario debe estar autenticado en el sistema y además debe haber alguna comisión de revisión en la base de datos.	
Entrada/ Pasos de ejecución: Pasos de ejecución: Seleccionar en el menú de navegación la opción “Comisiones”, el sistema mostrará una tabla con las comisiones y se debe seleccionar la que se desea visualizar.	
Resultado esperado: Se mostrarán los integrantes de la comisión de revisión.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#31. Caso de prueba de aceptación#7.

Caso de prueba de aceptación	
Código: HU3_CP3	**Historia de usuario:** 3
Nombre: Editar comisión de revisión	
Condiciones de ejecución: El usuario debe estar autenticado con el rol de Administrador del Sistema y además debe haber alguna comisión insertada en la base de datos.	
Entrada/ Pasos de ejecución: Entrada: Los datos de la comisión re revisión que el usuario quiere modificar. Pasos de ejecución: Seleccionar en el menú de navegación la opción “Comisiones”, el sistema mostrará una tabla con las comisiones y se debe seleccionar la que se desea editar, se mostrarán los integrantes y se eliminará o se añadirá los integrantes deseados, luego el sistema mostrará un mensaje de confirmación y además la comisión modificada en una tabla.	

Resultado esperado: La comisión de revisión es editado correctamente.
Evaluación de la prueba: Prueba satisfactoria

Tabla#32. Caso de prueba de aceptación#8.

Caso de prueba de aceptación	
Código: HU3_CP4	**Historia de usuario:** 3
Nombre: Eliminar comisión de revisión	
Condiciones de ejecución: El usuario debe estar autenticado con el rol de Administrador del Sistema y además debe haber alguna comisión de revisión insertada en la Base de Datos.	
Entrada/ Pasos de ejecución: Pasos de ejecución: Para eliminar una comisión se debe seleccionar del menú de navegación la opción “Comisiones” y luego eliminar la comisión deseada de la tabla.	
Resultado esperado: la comisión de revisión es eliminada correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#33. Caso de prueba de aceptación#9.

3.8 Análisis de los resultados de las pruebas.

Al concluir cada una de las iteraciones planificadas para el desarrollo de la propuesta de solución, fueron realizadas las pruebas pertinentes para realizar las entregas pactadas con el cliente, la Tabla 53 muestra el resultado de dichas pruebas en cada una de las iteraciones.

	Iteración 1	Iteración 2	Iteración 3	Iteración 4
Cantidad	12	7	5	0

Tabla#34. Cantidad de No Conformidades detectadas por iteración.

Todas las no conformidades fueron resueltas lo cual valida en cierto grado la calidad de la propuesta de solución, logrando una mayor satisfacción por parte del cliente.

3.9 Conclusiones.

En el desarrollo del capítulo se generaron artefactos de vital importancia para llevar a cabo la implementación de la propuesta de solución, tal es el caso de las tarjetas CRC y el DER, las primeras permitieron detallar la colaboración entre clases así como las responsabilidades de cada una, elementos que facilitaron la definición e implementación de las clases y funcionalidades por parte de los programadores del sistema; mientras que el DER permitió una mejor comprensión del modelo de datos. Se describió la arquitectura que presenta la propuesta de solución para un mejor entendimiento de la organización y distribución de los componentes del sistema. Al mismo tiempo se presenta un estudio de los patrones que posibilitaron un apoyo en cuanto al diseño del sistema. Se presentaron los estilos y

estándares más significativos utilizados durante el proceso de codificación, para facilitar el entendimiento del código. Se plantea la estrategia de pruebas seguida por el equipo de desarrollo, así como los resultados de las pruebas de aceptación, lo cual permitió brindar al cliente un *software* que satisface sus necesidades.

Conclusiones generales y Recomendaciones.

Conclusiones generales:

La culminación de la presente investigación permitió arribar a las siguientes conclusiones:

- Se implementó una herramienta multiplataforma que puede ser desplegada en los navegadores actuales sin restricciones.
- La solución permitió automatizar un proceso de gestión mostrando eficiencia en cuanto a tiempo de ejecución de los eventos investigativos y almacenamiento de la información que estos generan.
- Durante el proceso de validación de la herramienta fueron corregidas un total de 24 no conformidades detectadas durante las 4 iteraciones realizadas, las cuales influían negativamente en la usabilidad para los usuarios finales.
- El aporte social de la solución fue muy significativo, dado que se obtuvo una herramienta que automatiza un proceso complejo, realizando un cambio en cuanto a organización, divulgación y almacenamiento de información que generan los eventos investigativos, lo cual influye positivamente en el aumento de la participación por parte de la comunidad universitaria, posibilitando un aumento notable en el nivel académico e investigativo de la institución.

Recomendaciones:

Tomando como base la investigación realizada y el análisis de los resultados obtenidos se recomienda:

- Incluir el uso de patrocinadores en la presente solución, de modo que permita la divulgación de la información referente a los eventos investigativos en sus páginas oficiales.
- Desarrollar una versión que aproveche las principales tendencias del desarrollo de software actual, como son el desarrollo de software para dispositivos móviles, de modo que permita a los participantes acceder desde estos dispositivos.

Referencias Bibliográficas

1. **Acimed. 2002.** Gestión de información, gestión del conocimiento y gestión de la calidad en las organizaciones. [En línea] 2002. [Citado el: 21 de 3 de 2014.] http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94352002000500004.
2. *Agile manifesto.* **Alliance, Agile. 2001.** 2001.
3. **Ávila, Raiko Pulido. 2011.** Componentes de reportes para la monitorización a servidores de bases de datos PostgreSQL. [En línea] 2011. [Citado el: 30 de 1 de 2014.] http://repositorio_institucional.uci.cu/jspui/jspui/handle/ident/TD_04404_11.
4. **Becerra, Luis, y otros. 2013.** Arquitectura Cliente Servidor. [En línea] 6 de noviembre de 2013. [Citado el: 4 de febrero de 2014.] http://prezi.com/ggdtbtt8evyo/arquitectura-cliente-servidor.
5. **Beck, Kent. 2008.** *Extreme Programming Explained: embrace change.* [ed.] 2da Edicion. 2da. Boston : s.n., 2008.
6. **Centro de Documentación de Joomla, Spanish! 2014.** Características Principales.Joomla!Spanish. [En línea] 2014. [Citado el: 5 de 3 de 2014.] http//ayuda.joomlaspanish.org/-preguntas-frecuentes-enlaces-directos-82/388-características-principales..
7. **Centro de Documentación Joomla! Spanish. 2012.** Preguntas Frecuentes. Características Principales. *Joomla! Spanish.* [En línea] 2012. [Citado el: 10 de febrero de 2014.] http://ayuda.joomlaspanish.org/-preguntas-frecuentes-enlaces-directos-82/388-caracteristicas-principales.
8. **Científica, Serie. 2014.** Serie Científica. [En línea] 2014. [Citado el: 7 de 2 de 2014.] http://publicaciones.uci.cu/index.php/SC/index.
9. **Conde, Jesus. 2013.** Videotutoriales.com. [En línea] 2013. [Citado el: 14 de 4 de 2014.] http://paper.li/f-1391876133Videotutoriales.com.
10. **CUJAE. 2014.** Página principal. [En línea] 2014. [Citado el: 7 de 2 de 2014.] http://cujae.edu.cu/eventos/altae/Web%20ALTAE%202013%20UV.html.
11. **Drupal, Hispano.** Drupal Hispano | Comunidad de usuarios de Drupal. [En línea] [Citado el: 23 de 3 de 2014.] http://drupal.org.es/.
12. **Eguiluz, Javier.** La librería jQuery (Introducción a AJAX). [En línea] [Citado el: 19 de 2 de 2014.] from: http://librosweb.es/ajax/capitulo_10/la_libreria_jquery.html..
13. **Fowler, Kent Beck y Martin. 2000.** *Planning Xtreme Programming.* [ed.] Addison Wesley. 1ra Edición . 2000.
14. **Gestión, Sistema de.** [En línea] [Citado el: 13 de 2 de 2014.] http://www.bsigroup.com.mx/es-mx/Auditoria-y-Certificacion/Sistemas-de-Gestion.

15. **Group, The PHP Documentation. 2014.** [En línea] 2014. [Citado el: 2 de 3 de 2014.] http://www.php.net/docs.php.
16. **Guerrero, Rafael Martínez. 2010.** PostgreSQL. *Portal en espanol sobre PostgreSQL.* [En línea] 10 de 2 de 2010. [Citado el: 11 de 4 de 2014.] http://www.postgresql.org.es/sobre_postgresql.
17. **Habana, Universidad de La. 2014.** Universidad de La Habana | Calendario. [En línea] 2014. [Citado el: 7 de 2 de 2014.] http://www.uh.cu/calendar-date.
18. **Highsmith, James. 2002.** *A. Agile software development ecosystems.* s.l. : AddisonWesley Professional, 2002.
19. **hinostroza, R. R. 2005.** Características de PHP. [En línea] 2005. [Citado el: 1 de 3 de 2014.] http://www.linuxcentro.net/linux/staticpages/index.php?page=CaracteristicasPHP.
20. **Icesi. 2014.** Universidad Icesi - Gestión de eventos. [En línea] 2014. [Citado el: 7 de 2 de 2014.] http://www.icesi.edu.co/eventos/index.php/index/index/index.
21. **INFORMÁTICOS, DEPARTAMENTO DE LENGUAJES Y SISTEMAS.** Ingeniería de Software II. *Patrón Decorator.* [En línea] [Citado el: 14 de 3 de 2014.] http://www.lsi.us.es/docencia/get.php?id=1379.
22. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000.** *El proceso unificado de desarrollo de software.* s.l. : Addison Wesley, 2000.
23. **Jeffries, Ron, Anderson, Ann and Hendrickson. 2001.** *Extreme Programming Installed.* 2001. ISBN 9780201708424.
24. **Kair M., J. 2003.** *La Biblia de Servidor Apache 2.* Madrid : Anaya Mu, 2003. ISBN 84-415-1468-2.
25. **Koskela, Lasse. 2008.** *TEST DRIVEN, Practical TDD and Acceptance TDD for Java Developers.* s.l. : Manning Publication Co., 2008. ISBN 1-932394-85-0.
26. **Kruchten, Philippe. 2003.** *Rational Unified Process, An Introduction.* 3ra Edición. 2003.
27. **Larman, Craig. 2003.** *UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado.* s.l. : Pearson Educación, 2003. ISBN 9788420534381.
28. **Letelier, Patricio y Penadés, Mª Carmen. 2006.** Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). [En línea] 15 de enero de 2006. [Citado el: 30 de enero de 2014.] http://www.cyta.com.ar/ta0502/b_v5n2a1.htm.
29. **Mezquia, Diosblel Marimón. 2013.** Perfil de Instalación de Portales Intranet para el Sistema Gestor de Contenidos Drupal 7. La Habana : Universidad de las Ciencias Informáticas, 2013.

30. **Molpeceres, Alberto. 2002.** Procesos de desarrollo: RUP, XP y FDD. [En línea] 2002. [Citado el: 15 de febrero de 2014.] http://www.willydev.net/descargas/articulos/general/cualxpfddrup.PDF.
31. **Muñiz, Dayron Luis Busto.** Módulo de monitorización del Sistema Único de Identificación Nacional de la República de Cuba. *Universidad de las Ciencias Informáticas.* [En línea] [Citado el: 21 de 1 de 2014.]
32. **Murphey, Rebecca.** *jQuery Fundamentals.* ISBN 9781446671405.
33. **Musciano, C. 1999.** *HTML. La guía completa.* Mexico : s.n., 1999. ISBN 970-10-2141X.
34. **MySQL.** [En línea] [Citado el: 8 de 4 de 2014.] http://www.gridmorelos.uaem.mx/~mcruz//cursos/miic/MySQL.pdf .
35. **Olmos, Javier Ernesto Vigueras.** *Sistema para la gestión de los procesos de integración en las soluciones de almacenes de datos.* La Habana : Universidad de las Ciencias Informáticas. 005.74-Vig-S.
36. **Oracle Corporation and/or its affiliates. 2013.** Bienvenido a NetBeans y www.netbeans.org. [En línea] 2013. https://netbeans.org/index_es.html.
37. **Pablo Calabria, Luis Piriz. 2003.** Universidad ORT Uruguay : Cátedra de Ingeniería de Software. *Metodologia XP.* [En línea] 2003.
38. **Pereda Ojeda, Maikel y Gago Martínez, Yudanis. 2008.** Análisis, Diseño e Implementación del Módulo Experticias . La Habana : Universidad de las Ciencias Informáticas, 2008.
39. **Pérez, J. E. 2009.** *Introducción a JavaScript.* 2009.
40. **Pérez, J. E. 2008.** *Introducción al CSS.* 2008.
41. **Petrovsky, Michele.** *Manual de Dynamic HTML.* s.l. : Madrid: McGraw-Hill Interamericana. ISBN 84-481-1481-7.
42. **Potencier, Fabien y Zaninotto, François. 2011.** *Symfony 1.4, la guía definitiva.* 2011.
43. **Pressman, Roger S. 2002.** *Ingeniería Del software un enfoque practico.* 5ta Edición. Madrid : MacGraw-Hill, 2002.
44. **RAE. 2001.** Diccionario de la lengua española. [En línea] 2001. [Citado el: 13 de 1 de 2014.] http://lema.rae.es/drae/?val=evento.
45. **Reyero, Jose. 2009.** Características de Drupal. [En línea] 14 de agosto de 2009. [Citado el: 10 de febrero de 2014.] http://drupal.org.es/caracteristicas.
46. **Rodero, Luis Merino, Ortuño Pérez, Miguel y López Fernández, Luis.** Extension del patron Observador para la integración de Componentes heterogéneos. [En línea] [Citado el: 1 de 3 de 2014.] http://gsyc.escet.urjc.es/~mortuno/articulos/extension.pdf.

47. **Rodríguez Corbea, Maité y Ordóñez Pérez, Meylín . 2007.** La Metodología XP Aplicable al Desarrollo del Software educativo en Cuba. [En línea] 2007. [Citado el: 24 de 3 de 2014.] http://repositorio_institucional.uci.cu//jspui/handle/ident/TD_0837_07.
48. **Rodríguez, Fran Gil. 2012.** *Experto en Drupal 7. Nivel Inicial. Curso de creación y gestión de portales web con Drupal 7.* s.l. : Forcontu S.L, 2012. ISBN-13 (Edición electrónica, PDF): 978-84-939410-3-1.
49. **Ronquillo García, Dayrene y Fuentes Cano, Yadira . 2010.** Guia de buenas prácticas para obtener mantenibilidad en bases de datos desarrolladas en PostgreSQL. [En línea] 2010. [Citado el: 15 de 3 de 2014.] http://repositorio_institucional.uci.cu/jspui/jspui/handle/ident/TD_03531_10.
50. **Software, Unidad Docente de Ingeniería de.** Patrones del Ganf of Four. *Facultad de Informática-Universidad Politécnica de Madrid.* [En línea] [Citado el: 25 de 3 de 2014.] http://is.ls.fi.upm.es/docencia/proyecto/docs/patrones_gof.pdf.
51. **Sym.posium. 2014.** Sym.posium - Eventos. [En línea] 2014. [Citado el: 7 de 1 de 2014.] http://um.sym.posium.com/#1-next.
52. **Tramullas, Jesús. 2010.** Drupal para bibliotecas y archivos. [En línea] 2010. [Citado el: 27 de 1 de 2014.] http://libros.metabiblioteca.org/handle/001/178. ISBN 978-84-613-9611-5.
53. **UCI, Catálogo Biblioteca.** Catálogo Biblioteca UCI en línea. *Módulo de monitorización del Sistema Único de Identificación Nacional de la República de Cuba.* [En línea] [Citado el: 23 de 1 de 2014.] http://catalogoenlinea.uci.cu/cgibin/koha/opac-detail.pl?biblionumber=13134.
54. **VANDYK, K. 2008.** *Pro Drupal Development.* s.l. : Apress, 2008. ISBN 9781430209904.
55. **Visual-Paradigm. 2014.** Visual-Paradigm. [En línea] 2014. [Citado el: 12 de 1 de 2014.] http://www.visual-paradigm.com/product/vpuml/.
56. **W3C, Oficina española. 2014.** [En línea] 2014. [Citado el: 1 de 3 de 2014.] http://www.w3c.es/Divulgacion/a-z/#h.
57. **Welicki, León.** El Patrón Singleton. [En línea] [Citado el: 23 de 3 de 2014.] http://msdn.microsoft.com/es-es/library/bb972272.aspx.
58. **Willard, W. 2002.** Fundamentos de programación HTML. [En línea] 2002. [Citado el: 11 de 3 de 2014.] http://catalogoenlinea.uci.cu/cgi-bin/koha/opac-detail.pl?biblionumber=746. ISBN 958-41-0266-4.

Glosario:

A continuación, en orden alfabético, se muestra el significado de algunos términos usados en este documento que pueden dificultar la compresión del mismo:

1. **Arquitectura:** En informática, estructura lógica y física de los componentes de un sistema o software.
2. ***Banners*:** Es un formato publicitario en Internet. Esta forma de publicidad online consiste en incluir una pieza publicitaria dentro de una página web. Prácticamente en la totalidad de los casos, su objetivo es atraer tráfico hacia el sitio web del anunciante que paga por su inclusión.
3. ***Blogger***: Es un servicio creado por *Pyra Labs* y adquirido por Google en el año 2003, que permite crear y publicar una bitácora en línea. Para publicar contenidos, el usuario no tiene que escribir ningún código o instalar programas de servidor o de scripting.
4. ***Bookmark* (marcadores)**: Conocidos también como Favoritos, son aquellos enlaces a páginas web que se han almacenado en el navegador web por su interés o para su posterior visualización.
5. **Caché:** Búfer especial de memoria que poseen los ordenadores. Funciona de una manera similar a como lo hace la memoria principal (RAM), pero es de menor tamaño y de acceso más rápido.
6. ***Chat*:** Intercambio de mensajes electrónicos a través de internet que permite establecer una conversación entre dos o varias personas, servicio de mensajería electrónica.
7. ***Copyright*:** Derecho de autor, es un conjunto de normas jurídicas y principios que afirman los derechos morales y patrimoniales que la ley concede a los autores.
8. **Despliegue:** Acto de pasar las distintas unidades de un cuerpo (cualquier índole) a los puestos que deben ocupar.
9. **Diagrama:** Un diagrama o gráfico es un tipo de esquema de información que representa datos numéricos tabulados.
10. ***E-mail*: Correo electrónico** (en inglés: *e-mail*), es un servicio de red que permite a los usuarios enviar y recibir mensajes (también denominados mensajes electrónicos o cartas **electrónicas**) mediante sistemas de comunicación electrónicos.
11. **Entidades:** Lo que constituye la esencia o la forma de una cosa.

12. **Estándar:** Sirve como tipo, modelo, norma, patrón o referencia. En la informática es un término muy general utilizado para referirse a especificaciones técnicas que definen y describen aspectos de la *World Wide Web*.
13. **Estrategia:** Arte de dirigir las operaciones de cualquier índole.
14. **Evento:** Suceso importante y programado, de índole social, académica, artística o deportiva.
15. **Factor:** Recursos o materiales que al ser combinados en la producción agregan valor a los bienes y servicios.
16. ***Fanspage*:** Página de divulgación o promoción de algún artículo o evento de cualquier índole.
17. **Flexibilidad:** Cualidad de flexible, adaptable.
18. **Flujo:** En informática flujo de trabajo**,** estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información que soporta las tareas y cómo se le hace seguimiento al cumplimiento de las tareas.
19. **Foros:** Técnica de comunicación entre un grupo de personas.
20. **Gestión:** Ejercicio de responsabilidades sobre un proceso. También se entiende por gestión al conjunto de trámites a realizar para resolver un asunto.
21. ***Gateway*:** En informática, un equipo para interconectar redes.
22. **Globalización:** Tendencia de los mercados y de las empresas a extenderse, alcanzando una dimensión mundial que sobrepasa las fronteras nacionales.
23. ***Host*:** En informática es un anfitrión que proveen servicios a un número de computadoras conectadas a una red.
24. **Indexar:** Registrar ordenadamente datos e informaciones, para elaborar su índice.
25. **Interacción:** Acción que se ejerce recíprocamente entre dos o más objetos, agentes, fuerzas, funcione.
26. **Interfaz:** En informática se utiliza para nombrar a la conexión física y funcional entre dos sistemas o dispositivos de cualquier tipo dando una comunicación entre distintos niveles.
27. ***Jabber***: Es un protocolo libre de mensajería instantánea, baso en un estándar robusto como XML (*Extensible Markup Language*) y gestionado por XMPP *Standards Foundation*.

28. **Licencia:** Puede ser un verbo, es decir una de las formas de conjugación del verbo licenciar que significa "dar permiso" y pero también puede ser un sustantivo en cuyo caso representa tanto al permiso en si como al documento que registra dicho permiso.
29. ***LiveJournal***: Es el nombre de un sitio de *weblog* que permite a los internautas mantener un periódico o diario en línea.
30. **Maquetado:** Es un montaje funcional, a menor "escala", con materiales pensados para resaltar, en su funcionalidad, la atención de aquello que, en su escala real, presentará como innovación, mejora o sencillamente el gusto de quien lo monta.
31. **Método:** Procedimiento que se sigue en las ciencias para hallar la verdad y enseñarla.
32. **Metodología:** Conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal.
33. **Módulo:** En informática es una parte autónoma de un programa de ordenador.
34. **Multihilo:** Cualidad para ejecutar eficientemente múltiples hilos de ejecución.
35. **Multiproceso:** Uso de múltiples procesos concurrentes en un sistema en lugar de un único proceso en un instante determinado.
36. **Nodo:** En informática, un nodo es un punto de intersección o unión de varios elementos que confluyen en el mismo lugar.
37. **Notificación:** Acción y efecto de notificar o informar, información o datos de cualquier índole.
38. **Núcleo:** En computación, se refiere al núcleo (o kernel) de un sistema operativo.
39. **Patrón:** Conjunto de reglas que pueden ser usadas para crear o generar entidades.
40. **Procedimiento Almacenado:** Un procedimiento almacenado o *stored procedure*, es un programa(o función) el cual esta físicamente almacenado en una base de datos. La exacta implementación de un *stored procedure* incluyendo lo que es y cómo es implementado varía de un manejador de bases de datos a otro.
41. **Protocolo:** Protocolo de red para la comunicación de datos a través de paquetes conmutados.
42. **Requisito:** Circunstancia o condición necesaria para algo.
43. **Robustez:** En informática es un parámetro para determinar la seguridad y fuerza del sistema.
44. **Rol:** Comportamiento que se espera de cada uno según su estatus.

45. **Servidor:** En informática, un servidor es un nodo que, formando parte de una red, provee servicios a otros nodos denominados clientes.
46. **Software:** Equipamiento lógico o soporte lógico de un sistema informático, que comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas
47. **Taxonomía:** Ciencia que trata de los principios, métodos y fines de la clasificación.
48. **Transición:** Acción y efecto de pasar de un modo de ser o estar a otro distinto.
49. **Validar:** Dar fuerza o firmeza a algo, hacerlo válido.
50. **Viabilidad:** Cualidad de viable. Pretende determinar la racionalidad de las vías desde este punto de vista determinado.
51. **Vulnerable:** Que puede ser herido o recibir lesión, física o moralmente.
52. ***Web*:** En informática, la *World Wide Web* (WWW) o Red informática mundial comúnmente conocida como la web**,** es un sistema de distribución de documentos de hipertexto o hipermedias interconectados y accesibles vía Internet.

Anexos:

1-Ranking de Agilidad de las Metodologías Agiles.

	CMM	ASD	CRYSTAL	DSDM	FDD	LD	SCRUM	XP
Sistema como algo cambiante	1	5	4	3	3	4	5	5
Colaboración	2	5	5	4	4	4	5	5
Características Metodología(CM)								
Resultados	2	5	5	4	4	4	5	5
Simplicidad	1	4	4	3	5	3	5	5
Adaptabilidad	2	5	5	3	3	4	4	3
Excelencia técnica	4	3	3	4	4	4	3	4
Prácticas de colaboración	2	5	5	4	3	3	4	5
Media CM	2.2	4.4	4.4	3.6	3.8	3.6	4.2	4.4
Media total	1.7	4.8	4.5	3.6	3.6	3.9	4.7	4.8

Tabla#35. Ranking de ¨agilidad¨(los valores más altos representan una mayor agilidad).

2-Fases y flujo de trabajo RUP

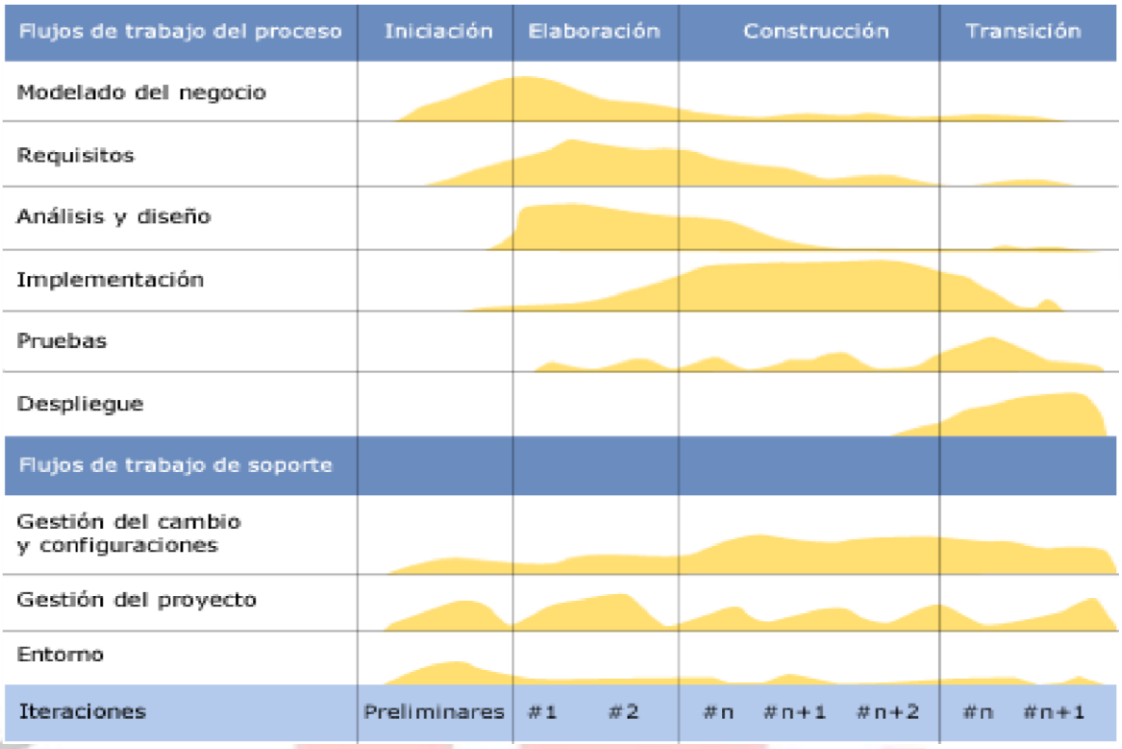

Ilustración 10. Fases y Flujo de Trabajo de RUP.

3- Casos de pruebas de Aceptación de las restantes historias de usuarios.

Caso de prueba de aceptación	
Código: HU4_CP1	**Historia de usuario:** 4
Nombre: Ver usuario.	
Condiciones de ejecución: El usuario debe estar autenticado correctamente en el sistema.	
Entrada/ Pasos de ejecución: Pasos de ejecución: Seleccionar en el menú de navegación la opción “Mi cuenta” donde se mostrarán todos los datos del usuario.	
Resultado esperado: Los datos del usuario son visualizados correctamente	

Evaluación de la prueba: Prueba satisfactoria

Tabla#36. Caso de prueba de aceptación#10.

Caso de prueba de aceptación	
Código: HU4_CP2	**Historia de usuario:** 4
Nombre: Editar usuario.	
Condiciones de ejecución: El usuario debe estar autenticado correctamente además de solo modificar el perfil que le corresponde.	
Entrada/ Pasos de ejecución: Pasos de ejecución: Seleccionar en el menú de navegación la opción “Mi cuenta” donde se mostrarán todos los datos del usuario, luego se debe seleccionar la opción “Editar”, el sistema mostrara una interfaz con los campos para introducir los datos deseados, luego se presiona el botón “Guardar” y el sistema mostrará un mensaje de confirmación además de los datos modificados.	
Resultado esperado: Los datos son editados correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#37. Caso de prueba de aceptación#11.

Caso de prueba de aceptación	
Código: HU4_CP3	**Historia de usuario:** 4
Nombre: Eliminar usuario.	
Condiciones de ejecución: El usuario debe estar autenticado con el rol de Administrador del Sistema.	
Entrada/ Pasos de ejecución: Pasos de ejecución: Seleccionar en el menú de navegación la opción “Usuarios” donde se mostrará una tabla con todos los usuarios, luego se debe seleccionar la opción “Editar” del usuario que se desea eliminar, el sistema mostrara una interfaz con los datos, luego se presiona el botón “Cancelar cuenta” y el sistema mostrará un mensaje de confirmación además de eliminar el usuario de la tabla.	
Resultado esperado: El usuario es eliminado correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#38. Caso de prueba de aceptación#12.

Caso de prueba de aceptación	
Código: HU5_CP1	**Historia de usuario:** 5
Nombre: Ver Calendario	
Condiciones de ejecución: El usuario debe estar autenticado correctamente en el sistema.	
Entrada/ Pasos de ejecución:	

Pasos de ejecución: Seleccionar en el menú principal la opción “Calendario” y el sistema mostrara un calendario con todos los eventos mensuales.
Resultado esperado: Los datos del calendario se visualizan correctamente.
Evaluación de la prueba: Prueba satisfactoria

Tabla#39. Caso de prueba de aceptación#13

Caso de prueba de aceptación	
Código: HU6_CP1	**Historia de usuario:** 6
Nombre: Ver estado del Proceso de revisión, aprobación y clasificación de trabajo	
Condiciones de ejecución: El usuario debe estar autenticado correctamente en el Sistema además de tener al menos un trabajo en concurso.	
Entrada/ Pasos de ejecución: Pasos de ejecución: Seleccionar en el menú de navegación la opción “Mis trabajos” donde se mostrará una tabla con todos los trabajos del usuario.	
Resultado esperado: El estado del trabajo se visualiza correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#40. Caso de prueba de aceptación#14.

Caso de prueba de aceptación	
Código: HU7_CP1	**Historia de usuario:** 7
Nombre: Ver búsqueda por metadatos	
Condiciones de ejecución: El usuario debe estar autenticado correctamente en el sistema además de haber realizado una búsqueda por metadatos	
Entrada/ Pasos de ejecución: Entrada: La entrada consiste en los datos de la búsqueda por metadatos. Pasos de ejecución: Seleccionar en el menú de navegación la opción “Búsqueda Avanzada” donde se mostrarán los campos que se desean filtrar.	
Resultado esperado: Los resultados de la búsqueda por metadatos son visualizados correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#41. Caso de prueba de aceptación#15.

Caso de prueba de aceptación	
Código: HU8_CP1	**Historia de usuario:** 8
Nombre: Permitir autenticar usuario.	
Condiciones de ejecución: El usuario debe estar registrado correctamente en el sistema.	
Entrada/ Pasos de ejecución: Entrada: Los datos del usuario que desea autenticarse.	

Pasos de ejecución: En la interfaz principal del sistema se deberán llenar los campos “usuario” y “contraseña”.
Resultado esperado: El usuario se autentica correctamente.
Evaluación de la prueba: Prueba satisfactoria

Tabla#42. Caso de prueba de aceptación#16.

Caso de prueba de aceptación	
Código: HU9_CP1	**Historia de usuario:** 9
Nombre: Permitir registrar usuario	
Condiciones de ejecución: El usuario no debe estar registrado en el Sistema.	
Entrada/ Pasos de ejecución: Entrada: La entrada consiste en los datos del usuario que desea ser registrado. Pasos de ejecución: En la interfaz principal seleccionar la opción “Crear nueva cuenta”, donde se mostraran los campos para introducir los datos, luego se deberá presionar el botón “Crear nueva cuenta”.	
Resultado esperado: el usuario es registrado correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#43. Caso de prueba de aceptación#17.

Caso de prueba de aceptación	
Código: HU10_CP2	**Historia de usuario:** 10
Nombre: Editar información sobre trabajo de evento	
Condiciones de ejecución: El usuario debe estar autenticado con el rol de Administrador del sistema y además debe haber algún evento insertado en la base de datos.	
Entrada/ Pasos de ejecución: Entrada: Los datos del trabajo que se quiere modificar. Pasos de ejecución: Para modificar los datos de un trabajo se debe seleccionar del menú principal la opción “Administrar trabajo”, luego seleccionar el trabajo deseado de una tabla mostrada por el sistema, luego presionar el botón “Modificar” y editar los campos deseados, al finalizar se debe presionar el botón “Guardar” y el sistema mostrará un mensaje de confirmación además de mostrar el trabajo modificado en la lista de trabajos.	
Resultado esperado: Los datos se editarán correctamente.	
Evaluación de la prueba: Prueba satisfactoria	

Tabla#44. Caso de prueba de aceptación#18.

4-Tareas de Ingeniería:

Tarea	
Número: *5*	**Número de HU:** *5*
Nombre: *Ver calendario.*	
Tipo de tarea: *Desarrollo*	**Estimación:** *3 día*
Fecha inicio: *17 de enero de 2014*	**Fecha fin:** *21 de enero de 2014*
Programador responsable: *Edelso A. Rojas, Jorge M. Soldevila*	
Descripción: *Ver el calendario del sistema.*	

Tabla#45. Tarea# 5.

Tarea	
Número: *6*	**Número de HU:** *7*
Nombre: *Ver búsqueda por metadatos.*	
Tipo de tarea: *Desarrollo*	**Estimación:** *3 día*
Fecha inicio: *22 de febrero de 2014*	**Fecha fin:** *24 de febrero de 2014*
Programador responsable: *Edelso A. Rojas, Jorge M. Soldevila*	
Descripción: *Visualizar resultados de la búsqueda por metadatos.*	

Tabla#46. Tarea# 6.

Tarea	
Número: *7*	**Número de HU:** *8*
Nombre: *Permitir autenticar usuario*	
Tipo de tarea: *Desarrollo*	**Estimación:** *3 día*
Fecha inicio: *27 de febrero de 2014*	**Fecha fin:** *29 de febrero de 2014*
Programador responsable: *Edelso A. Rojas, Jorge M. Soldevila*	
Descripción: *Permitir autenticar un usuario ya creado.*	

Tabla#47. Tarea# 7.

Tarea	
Número: *8*	**Número de HU:** *9*
Nombre: *Permitir registrar usuario*	
Tipo de tarea: *Desarrollo*	**Estimación:** *3 día*
Fecha inicio: *30 de febrero de 2014*	**Fecha fin:** *3 de febrero de 2014*
Programador responsable: *Edelso A. Rojas, Jorge M. Soldevila*	
Descripción: *Permitir registrar un usuario.*	

Tabla#48. Tarea# 8.

Tarea

Número: *9*	**Número de HU:** *4*
Nombre: *Ver usuario.*	
Tipo de tarea: *Desarrollo*	**Estimación:** *1 día*
Fecha inicio: *4 de febrero de 2014*	**Fecha fin:** *4 de febrero de 2014*
Programador responsable: *Edelso A. Rojas, Jorge M. Soldevila*	
Descripción: *Ver los datos de un usuario.*	

Tabla#49. Tarea# 9.

Tarea	
Número: *10*	**Número de HU:** *4*
Nombre: *Editar usuario.*	
Tipo de tarea: *Desarrollo*	**Estimación:** *1 día*
Fecha inicio: *5 de febrero de 2014*	**Fecha fin:** *5 de febrero de 2014*
Programador responsable: *Edelso A. Rojas, Jorge M. Soldevila*	
Descripción: *Editar los datos de un usuario.*	

Tabla#50. Tarea# 10.

Tarea	
Número: *11*	**Número de HU:** *4*
Nombre: *Eliminar usuario.*	
Tipo de tarea: *Desarrollo*	**Estimación:** *1 día*
Fecha inicio: *6 de febrero de 2014*	**Fecha fin:** *6 de febrero de 2014*
Programador responsable: *Edelso A. Rojas, Jorge M. Soldevila*	
Descripción: *Eliminar un usuario.*	

Tabla#51. Tarea# 11.

Tarea	
Número: *12*	**Número de HU:** *2*
Nombre: *Enviar Notificación.*	
Tipo de tarea: *Desarrollo*	**Estimación:** *3 día*
Fecha inicio: *7 de febrero de 2014*	**Fecha fin:** *11 de febrero de 2014*
Programador responsable: *Edelso A. Rojas, Jorge M. Soldevila*	
Descripción: *Enviar notificación con información acerca de inicio de evento o estado de trabajos.*	

Tabla#52. Tarea# 12.

Tarea	
Número: *13*	**Número de HU:** *3*
Nombre: *Adicionar comisión de revisión.*	
Tipo de tarea: *Desarrollo*	**Estimación:** *2 día*
Fecha inicio: *12 de febrero de 2014*	**Fecha fin:** *13 de febrero de 2014*
Programador responsable: *Edelso A. Rojas, Jorge M. Soldevila*	
Descripción: *Adicionar una comisión de revisión a un evento.*	

Tabla#53. Tarea# 13.

Tarea	
Número: 14	**Número de HU:** 3
Nombre: Ver comisión de revisión.	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 14 de febrero de 2014	**Fecha fin:** 14 de febrero de 2014
Programador responsable: Edelso A. Rojas, Jorge M. Soldevila	
Descripción: Ver los datos de una comisión de revisión de un evento.	

Tabla#54. Tarea# 14.

Tarea	
Número: 15	**Número de HU:** 3
Nombre: Editar Comisión de revisión.	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 17 de febrero de 2014	**Fecha fin:** 17 de febrero de 2014
Programador responsable: Edelso A. Rojas, Jorge M. Soldevila	
Descripción: *Editar los datos de una comisión de revisión.*	

Tabla#55. Tarea# 15

Tarea	
Número: 16	**Número de HU:** 3
Nombre: Eliminar comisión de revisión.	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 18 de febrero de 2014	**Fecha fin:** 18 de febrero de 2014
Programador responsable: Edelso A. Rojas, Jorge M. Soldevila	
Descripción: Eliminar una comisión de revisión.	

Tabla#56. Tarea# 16.

Tarea

Número: 17	**Número de HU:** 6
Nombre: Ver estado del Proceso de revisión, aprobación y clasificación de trabajo	
Tipo de tarea: Desarrollo	**Estimación:** 5 día
Fecha inicio: 19 de febrero de 2014	**Fecha fin:** 26 de febrero de 2014
Programador responsable: Edelso A. Rojas, Jorge M. Soldevila	
Descripción: Ver el estado del trabajo.	

Tabla#57. Tarea# 17.

Tarea	
Número: 18	**Número de HU:** 10
Nombre: Editar información sobre trabajo	
Tipo de tarea: Desarrollo	**Estimación:** 3 día
Fecha inicio: 27 de febrero de 2014	**Fecha fin:** 4 de marzo de 2014
Programador responsable: Edelso A. Rojas, Jorge M. Soldevila	
Descripción: Editar la información referente a un trabajo previamente registrado por un usuario del sistema en caso de algún error previo.	

Tabla#58. Tarea# 18.

5- Estructura del patrón *Singleton*:

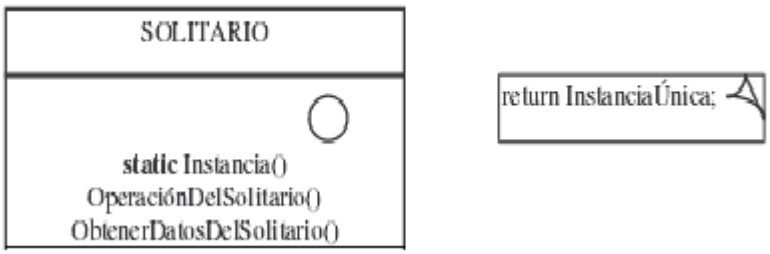

Ilustración 11. Estructura del patrón Singleton.

6- Gestor de Cambios en el Patrón *Observer*:

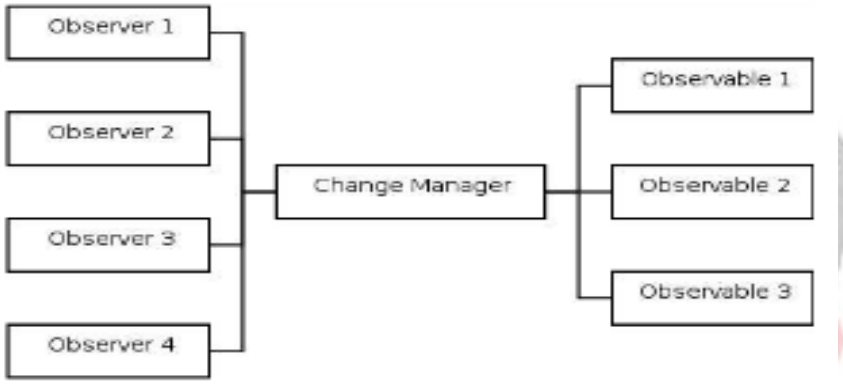

Ilustración 12. Gestor de Cambios en el Patrón Observer.

7- Estructura del patrón *Decorator*.

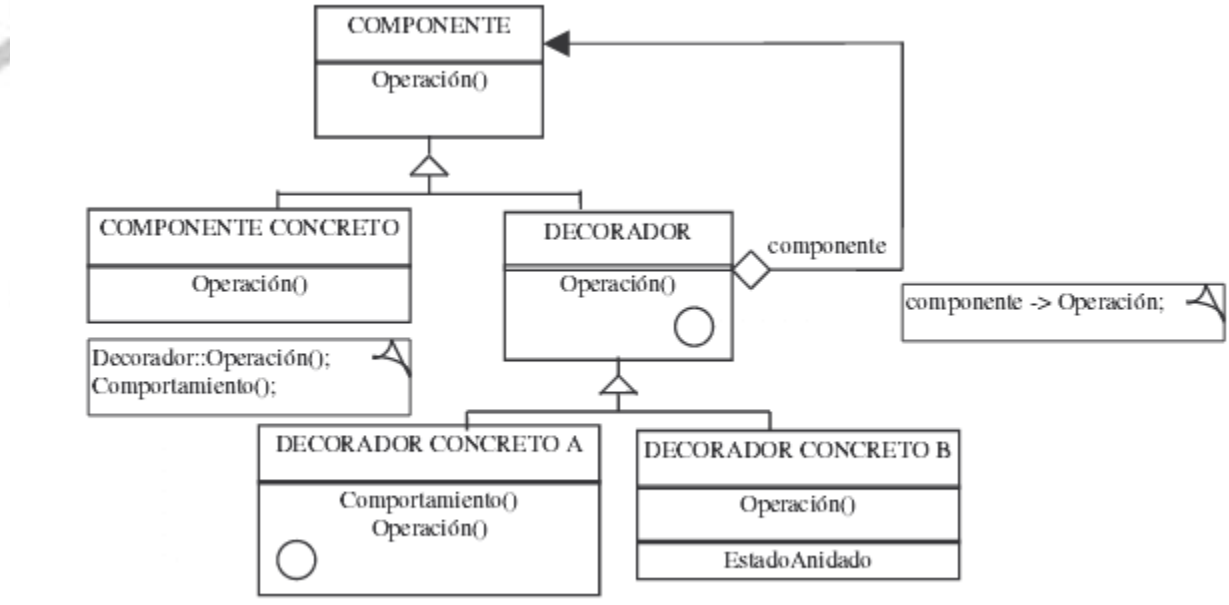

Ilustración 13. Estructura del patrón Decorator.

8- Estructura del patrón *Observer*.

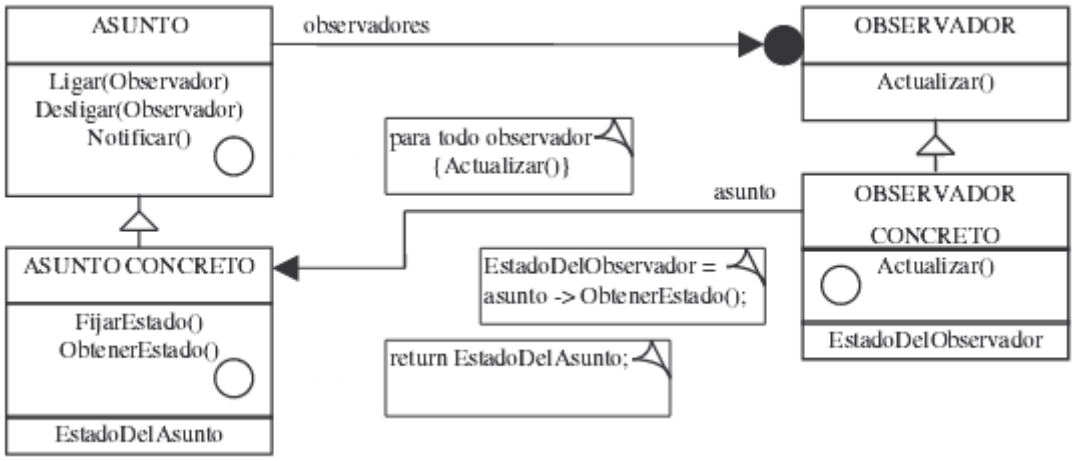

Ilustración 14. Estructura del Patrón Observer.