Universidad Central "Marta Abreu" de Las Villas

Facultad de Matemática, Física y Computación

Carrera Ingeniería Informática

TRABAJO DE DIPLOMA

Sistema para la gestión de la información del Departamento de Proyectos y Eventos de la Universidad Central "Marta Abreu" de Las Villas

Autor: Yandy Reyes Toledo

Tutor: MSc. Frank Reyes García

Santa Clara, 2016

Dedicatoria

A mi familia por estar siempre depositando todo su apoyo y su confianza en mí.

A mis amigos y vecinos que me apoyaron a lo largo de toda mi carrera.

Yandy Reyes Toledo

Agradecimientos

A mi familia.

A mis compañeros de grupo que me ayudaron y apoyaron hasta el final.

A mi tutor Frank Reyes García .

Yandy Reyes Toledo

Resumen

Dados los avances tecnológicos actuales se incrementa cada día el uso de las Tecnologías de la Información y las Comunicaciones en todas las ramas de la industria y el conocimiento, es por ello que también en este sentido dan la posibilidad de crear espacios informativos e interactivos que pueden ser constantemente enriquecidos con nuevos datos. Además se garantiza una mejor gestión de la información. El Departamento de Proyectos y Eventos de la Universidad Central "Marta Abreu" de Las Villas, el cual se encarga de facilitar la gestión institucional de los proyectos y eventos en la universidad, identificando oportunidades, áreas de desarrollo, iniciativas y líderes en la comunidad académica, en correspondencia con las estrategias de desarrollo del territorio. La presente investigación se realiza en dicho departamento y tiene como objetivo desarrollar una herramienta para la gestión de la información manejada por dicho departamento. La herramienta será capaz de evitar la entrada de datos erróneos o repetidos, además de facilitar la búsqueda de información. Dicha herramienta va a estar desarrollada usando el framework Symfony y el Sistema Gestor de Base de Datos PostgreSQL.

Abstract

Given current technological advances every day increases the use of Information Technology and Communications in all branches of industry and knowledge, which is why we also in this regard give us the possibility to create news programs and interactive they can be constantly enriched with new data. The Department of Projects and Events at the Central University "Marta Abreu" of Las Villas, which is responsible for facilitating institutional management of projects and events in the university, identifying opportunities, development areas, initiatives and leaders in the academic community, in line with the development strategies of the territory. This research is done in that department and aims to develop a tool for managing information managed by the department. This tool will be able to prevent the entry of erroneous or repeated data, and facilitate the search for information. This tool will be developed using the framework Symfony and Manager PostgreSQL database system.

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Introducción

Dados los avances tecnológicos actuales se incrementa cada día el uso de las Tecnologías de la Información y las Comunicaciones (TIC) en todas las ramas de la industria y el conocimiento, es por ello que también en este sentido dan la posibilidad de crear espacios informativos e interactivos que pueden ser constantemente enriquecidos con nuevos datos.

El Sistema de Gestión de Información aprovecha al máximo sus recursos de información en función de la mejora continua y de la toma de decisiones organizacional a todos los niveles jerárquicos desde la cúspide estratégica hasta la base operativa.

El Sistema de Gestión de Información tiene como objetivos:

1. Maximizar el valor y los beneficios derivados del uso de la información.
2. Minimizar el costo de adquisición, procesamiento y uso de la información.
3. Determinar responsabilidades para el uso efectivo, eficiente y económico de información.
4. Asegurar un suministro continuo de la información.

El Sistema de Gestión de Información aprovecha al máximo sus recursos de información en función de la mejora continua y de la toma de decisiones organizacional a todos los niveles jerárquicos desde la cúspide estratégica hasta la base operativa.

La siguiente investigación se desarrolla en el Departamento de Proyectos y Eventos (DPE) de la Universidad Central “Marta Abreu” de Las Villas (UCLV), proponiendo el desarrollo de un sistema para la gestión de su información. Para ello se analizan los antecedentes con respecto a antiguos sistemas existentes en dicho departamento.

El Departamento de Proyectos y Eventos (DPE) de la Universidad Central “Marta Abreu” de Las Villas (UCLV), el cual se encarga de facilitar la gestión institucional de los proyectos y eventos en la UCLV, era uno de los departamentos que pertenecía anteriormente a la Dirección de Relaciones Internacionales en la Universidad Central "Marta Abreu" de Las Villas (DRI-UCLV) y esta a su vez

respondía ante el Vicerrectorado de Investigación, Internacionalización y Postgrado (VRIIP), pero luego de una reestructuración el departamento pasó a responder directamente con el VRIIP. Luego de la reestructuración la DRI comenzó con un proceso de informatización de los procesos de sus departamentos permitiendo una mejor gestión y búsqueda de su información. El DPE también luego de la reestructuración vio la obligación de tener un sistema que le permitiera gestionar la información que maneja. En el DPE en este momento se maneja la información en ficheros Excel y en formato físico, por lo que no cuenta con un sistema informático, necesario para la gestión y difusión de la información de los proyectos, donaciones y eventos que se planifican en la misma.

Planteamiento de problema

En el Departamento de Proyectos y Eventos la información es manejada en ficheros Excel y en formato físico por lo que no cuentan con un sistema para gestionar la información. Existe dificultades en la búsqueda de información a la hora tomar una decisión por parte de los directivos. Existen errores al recopilar toda la información relevante. Para dar solución al presente problema de investigación se ha establecido el siguiente objetivo general.

Objetivo general:

Desarrollar un sistema para la gestión de la información del Departamento de Proyectos y Eventos de la UCLV.

A partir de un análisis del objetivo general se derivan los siguientes **objetivos específicos**:

1. Determinar los requisitos a informatizar a través del estudio el funcionamiento del departamento.
2. Diseñar una base de datos que permita almacenar los datos del departamento.
3. Diseñar una aplicación para gestionar la información del departamento.
4. Evaluar la validez del sistema mediante las pruebas de software de caja negra.

Estructura del Documento

En el Capítulo I se realiza una descripción del Departamento y de las herramientas a utilizar en el desarrollo de la herramienta.

En el Capítulo II se realiza una detallada descripción del modelo de negocio y los requisitos del Sistema. Se identifican reglas de negocio y los requisitos funcionales y no funcionales de la herramienta.

En el Capítulo III se realiza una descripción del funcionamiento de la herramienta en la etapa de diseño e implementación mediante los diagramas más significativos.

En el Capítulo IV se realiza una estimación del tiempo y el costo del desarrollo de la herramienta y además se realizan pruebas de caja negra para comprobar su funcionamiento.

Capítulo 1 “Fundamentación teórica”

1.1 Departamento de Proyectos y Eventos. (DPE)

El Departamento de Proyectos y Eventos de la UCLV es una organización de referencia en el país, que promueve y desarrolla una amplia colaboración nacional e internacional mediante proyectos y eventos, en correspondencia con las líneas de investigación universitaria y las estrategias de desarrollo del territorio. Dicho departamento realiza grandes aportes en moneda nacional y en divisa a la economía del país.

1.1.1 Misión y objetivos del DPE

El DPE tiene como misión fundamental facilitar la gestión institucional de los proyectos y eventos en la UCLV, identificando oportunidades, áreas de desarrollo, iniciativas y líderes en la comunidad académica, en correspondencia con las estrategias de desarrollo del territorio.

El DPE tiene como objetivos:

1. Estimular y facilitar la participación de nuestros científicos en proyectos de investigación e innovación tecnológica mediante la obtención de financiamiento (públicos y/o privados) para ponerlos en ejecución.
2. Impulsar y promover la innovación y el progreso tecnológico en los sectores productivos y el universitario a todos los niveles mediante la transferencia de los resultados de la investigación.
3. Lograr un amplio conocimiento y experiencia en la investigación de fuentes de financiación, la coordinación y la preparación de las ofertas, la negociación y el seguimiento de los proyectos y eventos.
4. Garantizar el apoyo a los investigadores en la preparación de propuestas de proyectos y eventos, su gestión económica-financiera y administrativa y la realización de los procedimientos de control interno.
5. Mantener y fortalecer las relaciones con los organismos de financiación y otras partes interesadas, tanto internas como externas, en los proyectos de cooperación y eventos.

6. Promover una estrategia territorial de cooperación internacional para maximizar sinergias y optimizar esfuerzos y recursos.(CAP, MEP-48)
7. Lograr niveles de gestión y organización superiores en los eventos científicos y académicos de alto impacto.
8. Promover las buenas prácticas en la gestión de proyectos y eventos entre la comunidad universitaria.

1.2 Tecnologías utilizadas en el desarrollo del entorno del Sistema.

En esta sección se describen las características necesarias en el proceso de selección de las tecnologías a utilizar en la solución propuesta y los beneficios que estas brindan para el desarrollo de aplicaciones Web.

1.2.1 Software Libre

Software libre es el software que respeta la libertad de los usuarios y la comunidad. A grandes rasgos, significa que los usuarios tienen la libertad de ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software. Es decir, el «software libre» es una cuestión de libertad, no de precio. Para entender el concepto, piense en «libre» como en «libre expresión», no como en «barra libre». En inglés a veces decimos «libre software», en lugar de «free software», para mostrar que no queremos decir que es gratuito(Chavarría, 2011).

El software libre promueve estas libertades porque todos merecen tenerlas. Con estas libertades, los usuarios (tanto individualmente como en forma colectiva) controlan el programa y lo que este hace. Cuando los usuarios no controlan el programa, decimos que dicho programa «no es libre», o que es «privativo». Un programa que no es libre controla a los usuarios, y el programador controla el programa, con lo cual el programa resulta ser un instrumento de poder injusto.

Un programa es software libre si los usuarios tienen las cuatro libertades esenciales:

- La libertad de ejecutar el programa como se desea, con cualquier propósito (libertad 0).
- La libertad de estudiar cómo funciona el programa, y cambiarlo para que haga lo que usted quiera (libertad 1). El acceso al código fuente es una condición necesaria para ello.

- La libertad de redistribuir copias para ayudar a su prójimo (libertad 2).
- La libertad de distribuir copias de sus versiones modificadas a terceros (libertad 3). Esto le permite ofrecer a toda la comunidad la oportunidad de beneficiarse de las modificaciones. El acceso al código fuente es una condición necesaria para ello. (Free Software Foundation, 2016)

1.2.2 ¿Por qué usar Software Libre?

A continuación se presentan algunas características que presenta el software libre que en gran medida son ventajosas debido a las libertades que ofrece. Algunas de ellas son:

- Libre: Existe libertad para usar, modificar, regalar o vender los programas de software libre.
- La copia es legal: Es legal repartir software libre a otras personas lo que evita en gran medida los problemas de la piratería. Si lo natural es compartir los programas con otras personas, con software libre es además legal.
- Abierto: se puede usar el código de los programas y modificarlo.
- Colaborativo: el modelo de desarrollo de software libre es colaborativo y participativo. Todo se puede modificar o criticar.(Adell & Bernabé, 2007)
- Ayuda: existen innumerables grupos de usuarios que se ayudan entre sí a través de Internet.
- Auditable: el software libre se puede inspeccionar al disponer de su código fuente.
- Corrección más rápida y eficiente de fallos: El funcionamiento e interés conjunto de la comunidad permite solucionar más rápidamente los fallos de seguridad en el software libre, algo que desgraciadamente en el software propietario es más difícil y costoso. Cuando se notifica a las empresas propietarias, éstas niegan inicialmente la existencia de dichos fallos por cuestiones de imagen, y cuando finalmente admiten la existencia de esos errores tardan meses hasta proporcionar los parches de seguridad(Mariño, 2006).
- Libertad de uso y redistribución: Las licencias de software libre actuales permiten la instalación el software tantas veces y en tantas máquinas como el usuario desee.

- Independencia tecnológica: El código fuente libre permite el desarrollo de nuevos productos sin la necesidad de desarrollar todo el proceso partiendo de cero.
- Fomento de la libre competencia al basarse en servicios y no licencias: Uno de los modelos de negocio que genera el software libre es la contratación de servicios de atención al cliente. Este sistema permite que las compañías que den el servicio compitan en igualdad de condiciones al no poseer la propiedad del producto del cual dan el servicio, lo que provoca un cambio vertical positivo en cuanto a atención al cliente y contratación de empleados, en contraposición a sistemas mayoritariamente sostenidos por la venta de licencias y desatención(Stallman, 2004).
- Soporte y compatibilidad a largo plazo: Más que una ventaja del software libre es una desventaja del software propietario. Al vendedor, una vez que ha alcanzado el máximo de ventas que puede realizar de un producto, no le interesa que sus clientes continúen con él. La opción es sacar un nuevo producto, producir software que emplee nuevas tecnologías solo para éste y no dar soporte para la resolución de fallos al anterior tratando de hacerlo obsoleto por todos los medios, pese a que el primero pudiera cubrir perfectamente las necesidades de muchos de los usuarios. La elección del software libre evade el problema.
- Formatos estándar: los formatos estándar permiten una interoperabilidad más alta entre sistemas, evitando incompatibilidades. Sistemas sin puertas traseras y más seguros: El acceso al código fuente permite a las empresas de seguridad de todo el mundo puedan auditar los programas, por lo que la existencia de puertas traseras es ilógica ya que se pondría en evidencia y contraviene el interés de la comunidad que es la que lo genera(LIBRE, 2010).

1.2.3 Principales lenguajes usados en el desarrollo de aplicaciones web

En la actualidad existen diferentes lenguajes de programación para el desarrollo de aplicaciones web, estos han ido surgiendo debido a las tendencias y necesidades de las plataformas. A continuación se pretende mostrar las ventajas y desventajas de los lenguajes más conocidos.

Lenguaje PHP

Es un lenguaje de programación utilizado para la creación de sitio web. PHP es un acrónimo recursivo que significa "PHP Hypertext Pre-processor", (inicialmente se llamó Personal Home Page). Surgió en 1995, desarrollado por PHP Group.

Es un lenguaje interpretado en el lado del servidor utilizado para la generación de páginas web dinámicas, embebidas en páginas de HyperText Markup Language (HTML) y ejecutadas en el servidor. PHP no necesita ser compilado para ejecutarse. Para su funcionamiento necesita tener instalado Apache o el servicio de información de internet con las bibliotecas de PHP. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas. Los archivos cuentan con la extensión (php) (Gallego, 2003).

Sintaxis:

La sintaxis utilizada para incorporar código PHP es la siguiente:

```
<?
$mensaje = "Hola";
echo $mensaje;
?>
```

También puede usarse:

```
<?php
$mensaje = "Hola";
echo $mensaje;
?>
```

Ventajas (J. D. Gutiérrez, 2004)**:**

- Muy fácil de aprender.
- Se caracteriza por ser un lenguaje muy rápido.
- Soporta en cierta medida la orientación a objeto. Clases y herencia.
- Es un lenguaje multiplataforma: Linux, Windows, entre otros.
- Capacidad de conexión con la mayoría de los manejadores de base de datos: MysSQL, PostgreSQL, Oracle, MS SQL Server, entre otras.
- Capacidad de expandir su potencial utilizando módulos.

- Posee documentación en su página oficial la cual incluye descripción y ejemplos de cada una de sus funciones.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Incluye gran cantidad de funciones.
- No requiere definición de tipos de variables ni manejo detallado del bajo nivel.

Desventajas (Martínez Echeverry, 2015)**:**

- Se necesita instalar un servidor web.
- Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.
- La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
- La programación orientada a objetos es aún muy deficiente para aplicaciones grandes.
- Dificulta la modularización.
- Dificulta la organización por capas de la aplicación.

Seguridad:

PHP es un poderoso lenguaje e intérprete, ya sea incluido como parte de un servidor web en forma de módulo o ejecutado como un binario Common Gateway Interface (término en inglés para «Interfaz de entrada común») separado, es capaz de acceder a archivos, ejecutar comandos y abrir conexiones de red en el servidor. Estas propiedades hacen que cualquier cosa que sea ejecutada en un servidor web sea insegura por naturaleza.

PHP está diseñado específicamente para ser un lenguaje más seguro para escribir programas CGI que Perl o C, y con la selección correcta de opciones de configuración en tiempos de compilación y ejecución, y siguiendo algunas prácticas correctas de programación.

Lenguaje Python

Es un lenguaje de programación creado en el año 1990 por Guido van Rossum, es el sucesor del lenguaje de programación ABC. Python es comparado habitualmente con Perl. Los usuarios lo consideran como un lenguaje más limpio para programar. Permite la creación de todo tipo de programas incluyendo los sitios web(Duque, 2011).

El código de Python no necesita ser compilado, por lo que se dice que es interpretado. Es un lenguaje de programación multiparadigma, lo cual fuerza a que los programadores adopten por un estilo de programación particular (Ramirez, 2010):

- Programación orientada a objetos.
- Programación estructurada.
- Programación funcional.
- Programación orientada a aspectos.

Sintaxis:

Ejemplo de una clase en Phyton:

```
def dibujar_muneco(opcion):
  if opcion == 1:
    C.create_line(580, 150, 580, 320, width=4, fill="blue")
    C.create_oval(510, 150, 560, 200, width=2, fill='PeachPuff')
```

Ventajas (Marzal & Gracia, 2003)**:**

- Libre y fuente abierta.
- Lenguaje de propósito general.
- Gran cantidad de funciones y bibliotecas.
- Sencillo y rápido de programar.
- Multiplataforma.
- Licencia de código abierto (Opensource).
- Orientado a Objetos.
- Portable.

Desventajas (Frittelli et al., 2013)**:**

- Lentitud por ser un lenguaje interpretado.

Lenguaje Ruby

Es un lenguaje interpretado de muy alto nivel y orientado a objetos. Desarrollado en el 1993 por el programador japonés Yukihiro "Matz" Matsumoto. Su sintaxis está inspirada en Phyton y Perl. Es distribuido bajo licencia de software libre (Opensource).

Ruby es un lenguaje dinámico para una programación orientada a objetos rápida y sencilla. Para los que deseen iniciarse en este lenguaje pueden encontrar un tutorial interactivo de ruby. Se encuentra también a disposición de estos usuarios un sitio con informaciones y cursos en español.

Sintaxis:

```
puts "hola"
```

Características:

- Existe diferencia entre mayúsculas y minúsculas.
- Múltiples expresiones por líneas, separadas por punto y coma ";".
- Dispone de manejo de excepciones.
- Ruby puede cargar bibliotecas de extensiones dinámicamente si el (Sistema Operativo) lo permite.
- Portátil.

Ventajas (Bykbaev, 2008)**:**

- Permite desarrollar soluciones a bajo Costo.
- Software libre.
- Multiplataforma.

Como se pudo observar todos estos lenguajes de programación para la web tienen sus ventajas y desventajas, pero se decidió usar el lenguaje PHP. A continuación se explican las razones del porque seleccionar PHP por encima de los demás lenguajes.

¿Porque usar PHP?

A la hora de seleccionar un lenguaje hay que tener en cuenta las cuatro grandes características: Velocidad, estabilidad, seguridad y simplicidad (Dondo, 2006).

- **Velocidad:** PHP no requiere de demasiados recursos del sistema por lo que no afecta la velocidad de ejecución, la cual es importante, ni crea demoras en la máquina. PHP se integra muy bien junto a otro software, especialmente bajo ambientes Unix, cuando se configura como módulo de Apache, está listo para ser utilizado.
- **Estabilidad:** La velocidad no sirve de mucho si el sistema se cae cada cierta cantidad de ejecuciones. Ninguna aplicación es 100% libre de bugs, pero teniendo de respaldo una increíble comunidad de programadores y usuarios es mucho más difícil para lo bugs sobrevivir. PHP utiliza su propio sistema de administración de recursos y dispone de un sofisticado método de manejo de variables, conformando un sistema robusto y estable.
- **Seguridad:** El sistema debe poseer protecciones contra ataques. PHP provee diferentes niveles de seguridad, estos pueden ser configurados desde el archivo .ini.
- **Simplicidad:** Se les debe permitir a los programadores generar código productivamente en el menor tiempo posible. Usuarios con experiencia en C y C++ podrán utilizar PHP rápidamente.

Otra característica a tener en cuenta seria la conectividad. PHP dispone de una amplia gama de bibliotecas, y agregarle extensiones es muy fácil. Esto le permite al PHP ser utilizado en muchas áreas diferentes, tales como encriptado, gráficos, XML y otras.

En la actualidad ya no es necesario el trabajo directo con el lenguaje PHP ya que existen algunos framework que le facilitan el trabajo a los desarrolladores.

1.2.4 Frameworks de desarrollo

Un frameworks es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje de scripting entre otros

software para ayudar a desarrollar y unir los diferentes componentes de un proyecto (J. J. Gutiérrez, 2014).

Un Software Framework es un diseño reusable de un sistema o subsistema. Está expresado por un conjunto de clases abstractas y el modo en que sus instancias colaboran para un tipo específico de software. Todos los frameworks de software son diseñados orientados a objetos(DEGLOVANNINI, 2005).

Los objetivos principales que persigue un framework son:

- Acelerar el proceso de desarrollo.
- Reutilizar código ya existente.
- Promover buenas prácticas de desarrollo como el uso de patrones.

Algunos framework gratuitos de trabajos:

Yii

Yii es un framework orientado a objetos de alto rendimiento basado en componentes. Utiliza el Patrón de diseño Modelo-Vista-Controlador (MVC). Generación compleja automática de WSDL, especificaciones y administración de peticiones Web Service(Ullman, 2013).

Algunas características de Yii según ("Features of Yii," 2016):

- Database Access Objects (DAO), query builder, Active Record y migración de base de datos.
- Integración con jQuery.
- Entradas de Formulario y validación.
- Widgets de Ajax, como autocompletado de campos de texto y demás.
- Soporte de Autenticación incorporado. Además soporta autorización vía role-based access control (RBAC) jerárquico.
- Personalización de aspectos y temas.

Django

Django es un framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como MVC. Fue desarrollado en origen para gestionar varias páginas orientadas a noticias de la World Company de Lawrence, Kansas, y fue liberada al público bajo una licencia BSD en julio de 2005;

el framework fue nombrado en alusión al guitarrista de jazz gitano Django Reinhardt(Holovaty & Kaplan-Moss, 2009).

Zend Framework (ZF)

ZF es un framework de código abierto para desarrollar aplicaciones web y servicios web con PHP 5. ZF es una implementación que usa código 100% orientado a objetos. En la estructura de los componentes de ZF; cada componente está construido con una baja dependencia de otros componentes. Esta arquitectura débilmente acoplada permite a los desarrolladores utilizar los componentes por separado(Vaswani, 2010).

Zend Framework utiliza Composer como gestor de dependencia de los paquetes; PHPUnit para poner a prueba todos los paquetes; y Travis CI como un servicio de integración continúa. ZF también sigue los estándares PHP - FIG, e incluye una implementación del PSR -7 para las interfaces de mensajes HTTP("Zend framework About," 2016).

CakePHP

CakePHP es un framework de desarrollo rápido de aplicaciones de código abierto en PHP. Inspirado en Rails, un framework para la construcción de sitios web que utilizan una base de datos como fuente de recursos, posee una infraestructura que tiene como finalidad permitir el desarrollo de aplicaciones web de manera ágil y estructurada, sin perder flexibilidad.

Entre las características más destacables de CakePHP según (Tupe & Cisneros, 2008):

- Arquitectura basada en el patrón MVC y orientada a objetos: define clases modelo, vista y controlador con funcionalidades básicas y de las cuales heredan todas las clases que se ajustan a este patrón y que son usadas en la aplicación construida con el framework.
- Una comunidad activa de usuarios: creada tras la publicación del framework en 2005 y que ha contribuido a mejorar el framework (a través de subproyectos específicos en CakeForge.Org) y difundir su uso.

- Licencia flexible: es distribuido bajo la licencia X11, más conocida entre los desarrolladores de software como MIT License.
- Compatible con PHP4 y PHP5: aunque en PHP4 se requiere especificar algunos parámetros de configuración adicionales en las clases a implementar.
- Operaciones básicas en base de datos (Creación, Obtención, Actualización y Borrado): estas operaciones están integradas para interacción con la base de datos y la simplificación de consultas.
- Estructura de aplicaciones (Application Scaffolding): permite al programador hacer uso de un conjunto de convenciones aplicables a la estructura de la base de datos de la aplicación y el framework se encarga de generar el código para la interacción a lo largo de todas las capas de la aplicación.
- Despachador de peticiones: permite acceder a la aplicación a través de URLs amigables y configurables.
- Incorporación de validaciones a lo largo del framework.
- Generación de plantillas de manera rápida y flexible: usando la sintaxis de PHP y con asistentes o helpers.
- Incorporación de asistentes de construcción de vistas: para la automatización de la generación de código en AJAX (Asynchronous JavaScript and XML), JavaScript, formularios HTML, entre otros.
- Componentes de seguridad, manejo de sesiones y de peticiones: que reúnen las mejoras prácticas estandarizadas por la industria del software.
- Listas de control de acceso flexibles: para gestionar el ingreso de usuarios a la aplicación construida con el framework.
- Verificación de ingreso de datos permitidos (Data Sanitization): permite determinar qué datos pueden ser ingresados y darle el formato adecuado a aquellos que no cumplen las reglas de validación.
- Almacenamiento en caché de las vistas: para acelerar la descarga de las páginas web.
- Trabaja en cualquier subdirectorio de un servidor web: requiere poca o nula configuración del servidor Apache donde se instalará.

El framework surge como un proyecto personal de Michal Tatarynowicz y a agosto de 2007 existe una versión estable (1.1.16.5612) y una versión de desarrollo (1.2.0.5427alpha).

Symfony

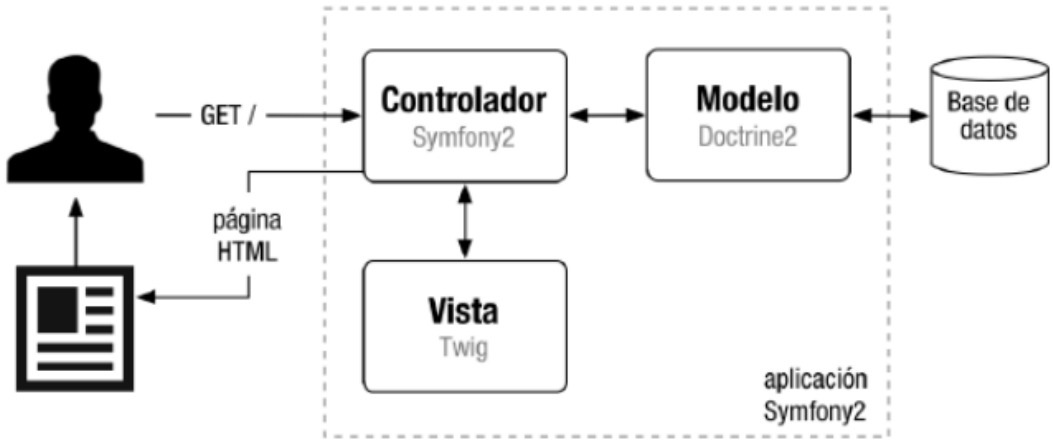

Figura 1.1 Estructura de Symfony

Symfony es un completo framework diseñado para optimizar el desarrollo de las aplicaciones web basado en el patrón Modelo Vista Controlador. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web(Eguiluz, 2013).

Symfony está desarrollado completamente en PHP 5.3. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows.(Fabien Potencier, 2013)

Fue diseñado para ajustarse a los siguientes requisitos (Potencier & Zaninotto, 2008):

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares).
- Independiente del sistema gestor de bases de datos. Su capa de abstracción y el uso de Propel, permiten cambiar con facilidad de SGBD en cualquier fase del proyecto.
- Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5.
- Sencillo de usar, aunque es preferible para el desarrollo de grandes aplicaciones Web que para pequeños proyectos.
- Aunque utiliza MVC (Modelo Vista Controlador), tiene su propia forma de trabajo en este punto, con variantes del MVC clásico como la capa de abstracción de base de datos, el controlador frontal y las acciones.
- Basado en la premisa de "convenir en vez de configurar", en la que el desarrollador sólo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.
- Una potente línea de comandos que facilitan generación de código, lo cual contribuye a ahorrar tiempo de trabajo.

Las características más comunes para el desarrollo de proyectos web están automatizadas en Symfony según (Ardissone, 2012):

- Permite la internacionalización para la traducción del texto de la interfaz, los datos y el contenido de localización.

- La presentación usa templates y layouts que pueden ser construidos por diseñadores de HTML que no posean conocimientos del framework.
- Los formularios soportan la validación automática, lo cual asegura mejor calidad de los datos en las base de datos y una mejor experiencia para el usuario.
- El manejo de caché reduce el uso de banda ancha y la carga del servidor.
- La facilidad de soportar autenticación y credenciales facilita la creación de áreas restringidas y manejo de seguridad de los usuarios.
- El enrutamiento y las URLs inteligentes hacen amigables las direcciones de las páginas de la aplicación.
- Las listas son más amigables, ya que permiten la paginación, clasificación y filtraje automáticos.
- Los plugins proveen un alto nivel de extensibilidad.
- La interacción con AJAX es mucho más sencilla.

1.2.5 ¿Porque usar Symfony?

El core de Symfony se basa en la integración de 4 productos muy potentes ya existentes en el mercado, que podríamos haber instalado de forma individual en nuestra aplicación:

Twig

Es un motor de plantillas "*rápido, seguro y flexible*", también propiedad de SensioLabs. Resuelve a la perfección la separación de la renderización de las **vistas** del patrón MVC. Permite herencia de plantillas, de forma que defines un *layout* general, y *bloques* de código html, similares a widgets, que son embebidos en cualquier parte del *layout*. Es extensible mediante el sistema de Bundles de Symfony, de forma que se pueden añadir nuevas funcionalidades muy fácilmente, como ejemplo, la manipulación de imágenes y generación de thumbnails con LiipImagineBundle, entre otros, o gestionar la personalización *temas* mediante LiipThemeBundle.

Doctrine

Es un ORM (mapeador de objetos-relacional) que facilitará mucho la vida a la hora de trabajar con la capa de persistencia en base de datos *(después de pillarle el truco a los archivos de definición de las entidades y las relaciones entre ellas, que no es inmediato)*, de mostrar/renderizar la información de una entidad y sus relacionadas en una página html, de transformarla en estructuras JSON o XML, de abstraernos el motor de base de datos a utilizar, de balancear entre las máquinas de un cluster de servidores maestros / esclavos según estemos actualizando o solo consultando, con drivers para conectaros a los motores más habituales (MySQL, PostgreSQL, Oracle, MS-SQL Server, SQL Lite, SAP Sybase SQL), y también a NoSQL como MongoDB(Pacheco, 2011).

Swiftmailer

Librería para el envío de correos electrónicos. Permite usar SMTP, sendmail, Postfix o un transporter personalizado, como por ejemplo GMail, de forma sencilla, soporta servidores que necesitan autentificación, fácil integración para envío de contenido HTML y texto plano alternativo, adjuntos, sistema de encolado de emails, registro de la traza de toda la comunicación con el servidor SMTP saliente, y más(Russell, 2016).

Monolog

Librería para registro de la actividad de una aplicación en ficheros de log, que implementa el estandar PSR-3. Define 9 niveles de criticidad de un mensaje *(log, debug, info, notice, warning, error, critical, alert, emergency)*, permite registrarlas en archivos de disco, base de datos, notificación por correo electrónico, definición de diferentes canales de logueo, y más. Por ejemplo, permite definir un canal por el cual, si la aplicación registra un mensaje de nivel *error* o superior, que lo guarde en un archivo de disco, y envíe un correo electrónico al equipo de soporte(Armand, 2014).

Ampliable

Este es uno de los **puntos fuertes**. En los pilares de la estructura de Symfony se encuentra la de organizar las funcionalidades en paquetes, o Bundles (en su propia terminología). Cuando se necesita algo que no está en el core lo más probable es que alguien haya tenido antes la misma necesidad, por lo que se puede consultar si existe algún Bundle que puedas instalar en tu aplicación que lo resuelva.

1.2.6 Sistema Gestor de Bases de Datos (SGBD)

Los sistemas gestores de bases de datos (SGBD), son un conjunto de programas que permiten crear y mantener una base de datos, asegurando su integridad, confidencialidad y seguridad(MySQL & MySQL, 2008).

Definimos un Sistema Gestor de Bases de Datos o SGBD, también llamado DBMS (*Data Base Management System*) como una colección de datos relacionados entre sí, estructurados y organizados, y un conjunto de programas que acceden y gestionan esos datos. La colección de esos datos se denomina Base de Datos o BD, (DB *Data Base*)(Holovaty & Kaplan-Moss, 2009).

Algunos gestores de base de datos:

MySQL

MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual GPL/Licencia comercial por Oracle Corporation y está considerada como la base datos open source más popular del mundo , y una de las más populares en general junto a Oracle y Microsoft SQL Server, sobre todo para entornos de desarrollo web(MySQL, 2000).

MongoDB

MongoDB (que proviene de «humongous») es la base de datos NoSQL líder y permite a las empresas ser más ágiles y escalables. Organizaciones de todos los tamaños están usando MongoDB para crear nuevos tipos de aplicaciones, mejorar la experiencia del cliente, acelerar el tiempo de comercialización y reducir costes.

Es una base de datos ágil que permite a los esquemas cambiar rápidamente cuando las aplicaciones evolucionan, proporcionando siempre la funcionalidad que los desarrolladores esperan de las bases de datos tradicionales, tales como índices

secundarios, un lenguaje completo de búsquedas y consistencia estricta(Banker, 2011).

Apache Derby

Apache Derby es un sistema gestor de base de datos relacional escrito en Java que puede ser empotrado en aplicaciones Java y utilizado para procesos de transacciones online. Tiene un tamaño de 2 MB de espacio en disco. Inicialmente distribuido como IBM Cloudscape, Apache Derby es un proyecto open source licenciado bajo la Apache 2.0 License. Actualmente se distribuye como Sun Java DB(Zikopolous, Baklarz, & Scott, 2005).

Algunas características:

- Su código pesa alrededor de 2000KB comprimido.
- Soporta cifrado completo, roles y permisos. Además posee SQL SCHEMAS para separar la información en una única base de datos y control completo de usuarios.
- Soporta internamente procedures, cifrado y compresión.
- Trae soporte multilenguaje y localizaciones específicas.

SQLite

SQLite Es un sistema de gestión de bases de datos relacional compatible con ACID, contenida en una relativamente pequeña biblioteca escrita en C. SQLite es un proyecto de dominio público.

Características:

• La biblioteca implementa la mayor parte del estándar SQL-92

• Varios procesos o hilos pueden acceder a la misma base de datos sin problemas

• Varios accesos de lectura pueden ser servidos en paralelo.

• Un acceso de escritura sólo puede ser servido si no se está sirviendo ningún otro acceso concurrentemente.

Ventajas:

- Rendimiento de base de datos.
- Interfaces.
- Costo.

Desventajas:

- Falta de Clave Foránea.
- Falta de documentación en español.

DB2

DB2: Es una marca comercial, propiedad de IBM, bajo la cual se comercializa el sistema de gestión de base de datos. La versión más actual es DB2 9, la cual utiliza XML como motor, además el modelo que utiliza es el jerárquico en lugar del modelo relacional que utilizan otros gestores.

Características:

- Permite el manejo de objetos grandes (hasta 2 GB).
- La definición de datos y funciones por parte del usuario, el chequeo de integridad referencial.
- SQL recursivo, soporte multimedia: texto, imágenes, video, audio; queries paralelos, commit de dos fases, backup/recuperación on−line y offline.

Ventajas:

- Permite agilizar el tiempo de respuestas de esta consulta.
- Recuperación utilizando accesos de sólo índices.
- Predicados correlacionados.
- Tablas de resumen.

- Tablas replicadas.
- Uniones hash.

Desventajas:

- Se tiene que ver con las aplicaciones que se tienen desarrolladas y las que se van a implementar.
- Influye en la elección.

PostgreSQL

Figura 1.2 PostgreSQL

PostgreSQL es un sistema para el manejo de bases de datos relacionales basado en Postgres, desarrollada en la Universidad de California en el departamento de Informática Berkeley. Postgres abrió el camino a muchos conceptos que sólo se volvieron disponibles en algunos sistemas de base de datos comerciales tiempo después. PostgreSQL es un código fuente abierto descendiente del código original de Berkeley.(Group, 2013)

Soporta en gran parte el estándar SQL y ofrece muchas características modernas como:

- Consultas complejas
- Llaves foráneas
- Triggers
- Updates
- Integridad transaccional

También, PostgreSQL puede extenderse por el usuario de muchas maneras, por ejemplo agregando nuevos:

- Tipos de datos

- Funciones
- Operadores
- Agregar funciones
- Idiomas procedurales

Debido a la licencia libre, PostgreSQL puede ser usado, modificado, distribuido gratuitamente por cualquiera para cualquier propósito, sea privado, comercial, o académico (Stonebraker & Kemnitz, 1991).

Algunas de sus principales características son según (Silberschatz et al., 2002):

Alta concurrencia

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

Amplia variedad de tipos nativos

PostgreSQL provee nativamente soporte para:

- Números de precisión arbitraria.
- Texto de largo ilimitado.
- Figuras geométricas (con una variedad de funciones asociadas).
- Direcciones IP (IPv4 e IPv6).
- Bloques de direcciones estilo CIDR.
- Direcciones MAC.
- Arrays.

A continuación las 10 razones más importantes por las que PostgreSQL es la base de datos que debemos usar:

1. Código fuente libre y de alta calidad

2. Licencia BSD - En pocas palabras, puedes hacer prácticamente lo que quieras con el producto, sin restricciones.
3. Soporte profesional tanto de la comunidad como de empresas especializadas.
4. Requerimientos de administración y mantenimiento relativamente bajos con respecto el resto de bases de datos comerciales
5. Fiabilidad y estabilidad legendarias
6. Rendimiento excelente
7. Diseñada para entornos con altos volúmenes de tráfico/transacciones
8. Extensible
9. Multiplataforma
10. Herramientas gráficas y de línea de comandos para diseñar nuestras bases de datos y administrarlas

1.2.7 Otras Tecnologías usadas en el diseño del sitio

Bootstrap

Bootstrap fue una herramienta originalmente desarrollada por (previamente) los ingenieros Mark Otto y Jacob Thorton como un intento de fomentar la utilización del mismo marco de desarrollo para minimizar las inconsistencias en el equipo de ingeniería.(Cochran, 2012)

La iniciativa de Bootstrap triunfó puesto que el equipo comenzó a trabajar más rápido, de forma más eficaz y con menos inconsistencias.

Bootstrap es una colección de varios elementos web personalizables y funciones completamente empaquetado en una sola herramienta. Cuando se diseña una web con Bootstrap, los desarrolladores pueden elegir qué elementos utilizar. Aún más importante, tienen la certeza de saber que los elementos que elijan no generarán conflictos entre ellos. Como si se tratase de un puzle, exceptuando que cada pieza del puzzle encaja perfectamente con las otras, sin importar la pieza que elija.

Los elementos personalizables de Bootstrap son una combinación de HTML, CSS y JavaScript. Bootstrap vive en una mejora continua. Se le han añadido una variedad de funcionalidades tales como responsabilidad 100% a dispositivos móviles.

JQuery

JQuery es una biblioteca gratuita de Javascript, cuyo objetivo principal es simplificar las tareas de creación de páginas web responsivas, acordes a lo estipulado en la Web 2.0, la cual funciona en todos los navegadores modernos. Por otro lado, se dice que jQuery ayuda a que nos concentremos de gran manera en el diseño del sitio, al abstraer por completo todas las características específicas de cada uno de los navegadores. Otra de las grandes ventajas de jQuery es que se enfoca en simplificar los scripts y en acceder/modificar el contenido de una página web. Finalmente, jQuery agrega una cantidad impresionante de efectos nuevos a Javascript, los cuales podrán ser utilizados en tus sitios Web(Chaffer & Swedberg, 2007).

1.3 Conclusiones Parciales

Con el desarrollo del Capítulo I hemos podido conocer más de cerca el funcionamiento del Departamento de Proyectos y Eventos y de las tecnologías a utilizar en el desarrollo del sistema.

Capítulo 2 Modelo del Negocio y Requisitos

En este capítulo se realiza una descripción del modelo de negocio y los requisitos a informatizar. Dentro de los cuales esta las reglas de negocio, actores y trabajadores del negocio, casos de uso del negocio y del sistema, requisitos funcionales y no funcionales, entre otros diagramas que son de ayuda para comprender como va a quedar el sistema a desarrollar.

2.1 Modelo de Negocio Departamento de Proyectos y Eventos

Según(Ivar Jacobson, 2000), un modelo de caso de uso del negocio describe los procesos de una empresa en términos de casos de uso del negocio y actores del mismo que se corresponden con los procesos del negocio y los clientes, respectivamente. El modelo de casos de uso del negocio presenta un sistema, desde la perspectiva de su uso, y esquematiza como proporciona valor a sus usuarios.

2.1.1 Reglas de negocio a considerar

RN1: Los gastos tienen que estar asociados a un proyecto.

RN2: El total de gastos no debe sobrepasar el total de presupuesto del proyecto.

RN3: El presupuesto anual no puede ser mayor que el presupuesto total.

2.1.2 Actores del Negocio

Los actores representan terceros fuera del sistema que colaboran con él. Los actores del sistema suelen corresponderse con los trabajadores del negocio y si algún actor del negocio va a interactuar con el sistema pasa también a ser actor del mismo.

Tabla 2.1 Actores del negocio

Actores del Negocio	Descripción
<Presidente del Comité Organizador>	Es la persona que presenta el evento para ser aprobado y en caso de serlo quien se apoya en el Departamento para ser asesorado.
<Jefe de Proyecto>	Es la persona que presenta el proyecto para ser aprobado y en caso de serlo

	quien se apoya en el Departamento para ser asesorado.

2.1.3 Diagrama de Caso de Uso del Negocio

Muestra los trabajadores del negocio y las entidades del negocio así como las asociaciones entre los mismos. Solo se tiene en cuenta "¿QUIÉN manipula QUÉ información?"¿QUIÉN? (trabajador del negocio identificado).¿QUÉ? (entidad del negocio identificado). Relaciones entre ellos (asociaciones).

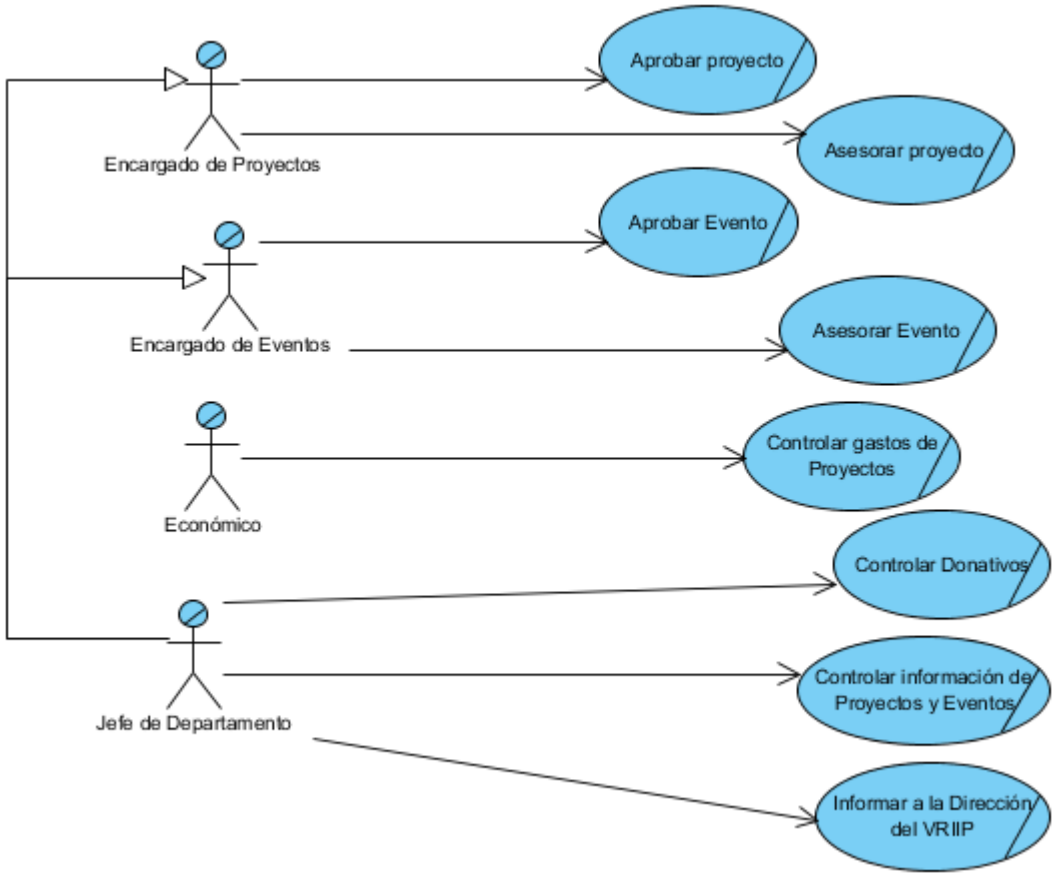

Figura 2.1 Diagrama de Caso de Uso de Negocio

2.1.4 Trabajadores del Negocio

En esta sección se hace referencia a los trabajadores del negocio ofreciendo una breve descripción de las funcionalidades correspondientes a cada uno, todo esto queda expresado en la Tabla 2.2.

Tabla 2.2 Trabajadores del negocio

Trabajadores del Negocio	**Descripción**
<Responsables de Proyectos>	Es el trabajador que se encarga de manipular toda la información referente a los proyectos y asesorar los mismos. Este trabajador agrupa todos los trabajadores del grupo de proyectos.
< Responsable de eventos >	Es el trabajador que se encarga de manipular toda la información referente a los eventos. Este trabajador agrupa todos los trabajadores del grupo de eventos.
<Jefe de Departamento>	Es el trabajador que se encarga de manipular toda la información referente a los donativos hechos a la UCLV. Además es el trabajador que se encarga del control total del departamento y tiene que mantenerse informado sobre todo lo referente a los donativos, proyectos y eventos. También tiene que informar a la Dirección del Vicerrectorado de Investigación, Internacionalización y Posgrado. (VRIIP).
<Económico>	Es el trabajador que se encarga del control de presupuesto y gastos de los proyecto.

2.1.5 Casos de Uso del negocio

En la Tabla 2.3 se describe el funcionamiento de algunos de los casos de usos más importantes del departamento.

Tabla 2.3 Casos de Uso del negocio

Caso de Uso del Negocio	**Descripción**
Aprobar proyecto	En este caso el responsable de proyectos es un intermediario entre el Jefe del proyecto y las organizaciones que se encargan de aprobar o denegar la ejecución del proyecto.
Asesorar proyecto	Lo realiza el responsable de proyectos y consiste en dar el adecuado seguimiento al proyecto para su correcta y segura ejecución.
Aprobar evento	En este caso el responsable de eventos es un intermediario entre el Presidente del Comité del evento y las organizaciones que se encargan de aprobar o denegar la ejecución del evento.
Asesorar evento	Lo realiza el responsable de eventos y consiste en apoyar al Presidente del Comité del evento para una correcta elaboración y ejecución del evento.
Recibir donativo	Lo realiza el Jefe de Departamento y consiste en mantenerse informado de todos

	los donativos realizados a la UCLV.
Informar a la Dirección del Vicerrectorado de Investigación, Internacionalización y Posgrado (VRIIP).	Lo realiza el Jefe de Departamento y consiste en informar a la Dirección del VRIIP sobre toda la información que necesiten acerca de los donativos proyectos y eventos.
Controlar información	Lo realiza el Jefe de Departamento y consiste en mantenerse informado acerca de todos los donativos, proyectos y eventos.
Controlar gastos de proyectos	Lo realiza el económico del departamento y consiste en llevar un control estricto de los gastos de los proyectos.

Nota: El departamento no es el que se encarga exactamente de la aprobación de los proyectos, pero se ve envuelto en el proceso.

2.1.6 Actores del Sistema

Tabla 2.4 Actores del Sistema

Actores del Sistema	Descripción
Encargado de Proyectos	Este actor se encarga de la gestión de toda la información referente a los proyectos, para ello necesita estar autenticado.
Encargado de Eventos	Este actor se encarga de la gestión de toda la información referente a los eventos, para ello necesita estar autenticado.
Jefe de Departamento	Este actor hereda las funcionalidades de los encargados del departamento, además gestiona la información de los donativos y tiene que mantenerse in formado sobre los proyectos y eventos para informar a la Dirección del VRIIP, para ello necesita estar autenticado.

Económico	Este actor se encarga de gestionar la información referente a los gastos de los proyectos, para ello necesita estar autenticado.
Jefe de Proyecto	Este actor puede modificar los datos de su proyecto y mantenerlo actualizado, para ello necesita estar autenticado.
Administrador	Este actor es el encargado de gestionar los usuarios que van a interactuar con el sistema, para realizar esa acción necesita estar previamente autentificado.

2.1.7 Requisitos

Los requisitos tienen como propósito fundamental guiar correctamente el desarrollo del sistema. Esto se consigue mediante una descripción de los requisitos del sistema detallada para que pueda llegarse a un acuerdo entre el cliente (incluyendo a los usuarios) y los desarrolladores sobre qué debe y qué no debe hacer el sistema.

Existen dos tipos de requisitos *los requisitos funcionales (RF) y los requisitos no funcionales (RNF) (Roger S. Pressman, 2010).* Un requisito funcional es una funcionalidad que debe tener el sistema, siendo identificado mediante los casos de uso del sistema. Por otro lado los requisitos no funcionales son propiedades o cualidades, se definen luego conocer cuáles van a ser los requisitos funcionales, es decir una vez que se conozca lo que el sistema debe hacer, se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser.

2.1.7.1 Requisitos funcionales

RF1: Mostrar información del departamento

RF2: Mostrar lista de Proyectos Nacionales en curso

RF3: Mostrar lista de Proyectos Internacionales en curso

RF4: Mostrar lista de Eventos del año actual

RF5: Mostrar información de Proyecto Nacional

RF6: Mostrar información de Proyecto Internacional

RF7: Gestionar Proyectos

RF7.1: Listar Proyectos

RF7.2: Editar Proyecto

RF7.3: Insertar Proyecto

RF7.4: Mostrar información del Proyecto

RF8: Gestionar Eventos

RF8.1: Insertar Evento

RF8.2: Listar Eventos

RF8.3: Editar Evento

RF8.4: Mostrar información del Evento

RF9: Gestionar Donativos

RF9.1: Insertar Donativo

RF9.2: Listar Donativos

RF9.3: Editar Donativo

RF9.4: Mostrar información del Donativo

RF10: Gestionar Gastos

RF10.1: Insertar gasto

RF10.2: Editar gasto

RF10.3: Listar gasto

RF10.4: Mostrar información del Donativo

RF11: Gestionar Usuario

RF11.1: Insertar Usuario

RF11.2: Listar Usuarios

RF11.3: Editar Usuario

RF11.4: Mostrar información del Usuario

RF11.5: Eliminar Usuario

RF12: Actualizar Normativas

RF13: Gestionar Comité Organizador

RF13.1: Listar Comité Organizador

RF13.2: Editar Comité Organizador

RF14: Gestionar Comité Científico

RF14.1: Listar Comité Científico

RF14.2: Editar Comité Científico

RF14: Gestionar Tema

RF14.1: Listar Tema

RF14.2: Editar Tema

2.1.7.2 Requisitos no funcionales

RNF1: Usabilidad.

La aplicación podrá ser utilizada desde cualquier lugar que tenga acceso a la red universitaria, todos tendrán acceso a la información pública y los trabajadores autorizados, tendrán un nivel de acceso correspondiente la información.

RNF2: Diseño.

Las interfaces se diseñaran de manera tal que le facilite el acceso al usuario permitiéndole entender con mayor claridad el contenido que se desea dar a conocer.

RNF3: Rendimiento.

El tiempo de respuesta no debe exceder los 3 segundos ante las solicitudes del usuario.

RNF4: Restricciones de diseño.

Sistema de Gestión de Bases de Datos: Se utilizará como gestor de base de datos PostgreSQL.

Lenguaje de Programación: Se necesita PHP en su versión 5.5.12 o superior.

RNF5: Ayuda y Documentación.

Manuales de usuario: Se brindará un manual de ayuda que explicará cómo trabajar de forma adecuada con el sistema.

RNF6: Interfaz

RNF6.1: Elementos gráficos distintivos: La interfaz contará con elementos gráficos representativos y menús desplegables que faciliten su utilización.

RNF6.2: Logo de la empresa: La interfaz debe tener incluido el logo de la empresa como un término de identificación.

2.1.9 Diagrama de caso de uso del sistema

En la Figura 2.2 se muestra el diagrama de Casos de Uso del Sistema correspondiente al sistema a desarrollar.

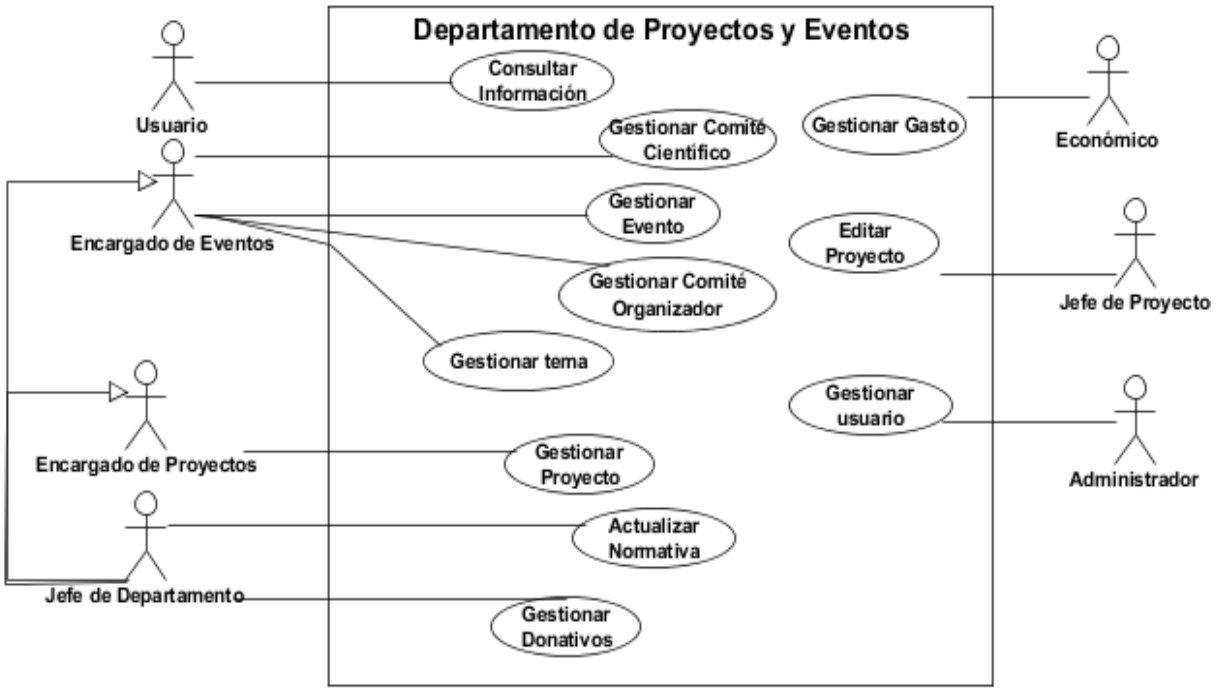

Figura 2.2 Diagrama de caso de Uso del Sistema

2.1.11 Descripción de los Casos de Uso del Sistema

A continuación se describen

Tabla 2.5 Caso de Uso del Sistema Insertar Proyecto

Caso de Uso del Sistema	Insertar Proyecto
Actores	Encargado de Proyectos, Jefe Departamento
Propósito	Insertar un nuevo proyecto.
Resumen	El caso de uso se inicia cuando el Encargado de Proyectos adiciona un

	nuevo proyecto
Precondiciones	El Encargado de Proyectos debe estar autenticado en el sistema
Flujo Normal de Procesos	
Acción del Actor	Respuesta del sistema
1. El Encargado de Proyectos selecciona la opción Insertar Proyecto.	
	2. El sistema muestra una ventana con el formulario para rellenar los datos del proyecto.
3. El Encargado de Proyectos inserta los datos.	
4. El Encargado de Proyectos selecciona la opción Guardar.	
	5. El sistema verifica que no haya ningún campo vacío y que los datos sean correctos.
	6. El sistema guarda los datos en la base de datos.
	7. El sistema muestra la página del listado de proyectos e informa que se guardaron los datos.
Otra sección	

1. El Encargado de Proyectos escoge la opción Cancelar.	
	2. El sistema lo lleva de regreso al listado de proyectos.
Flujo Alternativo de Eventos	
Mensaje de error por campos vacíos o incorrectos	
4. El Encargado de Proyectos selecciona la opción Guardar.	
	5. El sistema verifica que no haya ningún campo vacío y que los datos sean correctos.
	6. El sistema muestra un mensaje de error en el campo vacío o incorrecto.

Tabla 2.6 Caso de Uso del Sistema Insertar Evento

Caso de Uso del Sistema	Insertar Evento
Actores	Encargado de Eventos, Jefe Departamento
Propósito	Insertar un nuevo evento en el sistema
Resumen	El caso de uso se inicia cuando el Encargado de Eventos adiciona un nuevo evento
Precondiciones	El Encargado de Eventos debe estar autenticado en el sistema

Flujo Normal de Procesos	
Acción del Actor	Respuesta del sistema
1. El Encargado de Eventos selecciona la opción Insertar.	
	2. El sistema muestra una ventana con el formulario para rellenar los datos del evento.
3. El Encargado de Eventos inserta los datos generales del evento.	
4. El Encargado de Eventos selecciona la opción Guardar.	
	5. El sistema verifica que no haya ningún campo vacío y que los datos sean correctos.
	6. El sistema guarda los datos en la base de datos.
	7. El sistema muestra la página de Nuevo Miembro del Comité Organizador e informa que se guardaron los datos.
Otra sección	
1. El Encargado de Proyectos escoge la opción Cancelar.	
	2. El sistema lo lleva de regreso al

	listado de proyectos.
Flujo Alternativo de Eventos	
Mensaje de error por campos vacíos o incorrectos(para todos los guardar)	
4. El Encargado de Proyectos selecciona la opción Guardar.	
	5. El sistema verifica que no haya ningún campo vacío y que los datos sean correctos.
	6. El sistema muestra un mensaje de error en el campo vacío o incorrecto.

2.1.10 Diagrama de paquetes y sus relaciones

Los diagramas de paquetes muestran cómo se dividen en agrupaciones lógicas un sistema y ayudan a la fácil comprensión de la estructura de un software. Las agrupaciones lógicas que presenta el sistema desarrollado son:

Controller: En este paquete están las clases que se encargan de controlar la lógica del sistema para su correcto funcionamiento.

Entity: En este paquete están las clases que se convertirán en tablas en la base de datos.

Resources: En este paquete se encuentran las rutas y las vistas del sistema.

Form: En este paquete se encuentran las clases que luego se convertirán en formularios.

Repository: En este paquete se encuentran las clases que contienen las consultas a la base de datos.

En la Figura 2.3 se muestra el diagrama de paquetes correspondiente al sistema.

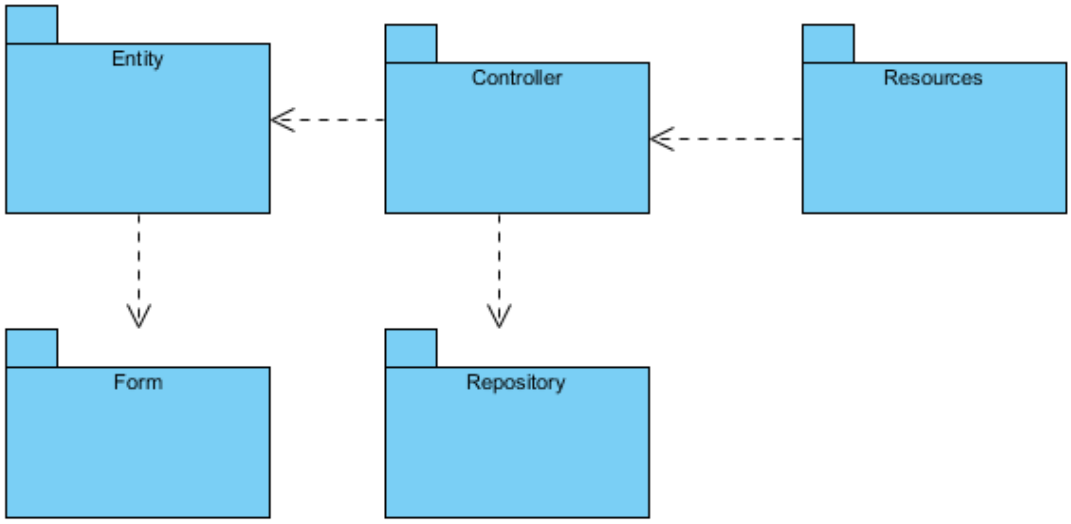

Figura 2.3 Diagrama de Paquetes

2.2 Conclusiones parciales

A lo largo del Capítulo 2 se ha podido observar cómo va a ser el funcionamiento del sistema a través del estudio del funcionamiento del departamento y de algunos diagramas que nos son de ayuda a la hora entender cómo va a ser la estructura interna de la misma.

Capítulo 3 Descripción de la propuesta de solución

En el Capítulo 3 se describe la propuesta de solución al problema de investigación. Para ello se representa cómo va a ser la interacción de las clases del sistema mediante algunos diagramas como son: el diagrama de clases del diseño diagrama de secuencia, diseño de la base de datos, diagrama de despliegue y además se explica el tratamiento de errores.

3.1 Arquitectura del Sistema

Este sistema esta implementado en Symfony en su versión 2.8.3 que utiliza una arquitectura MVC.

El término MVC proviene de tres palabras que hoy en día se utilizan mucho dentro del ambiente de desarrollo de software: Model – View – Controller, lo que sería en castellano Modelado, Vista y Controlador. Esta arquitectura permite dividir nuestras aplicaciones en tres grandes capas:

Vista: Todo lo que se refiera a la visualización de la información, el diseño, colores, estilos y la estructura visual en sí de nuestras páginas.

Modelado: Es el responsable de la conexión a la base de datos y la manipulación de los datos mismos. Esta capa está pensada para trabajar con los datos como así también obtenerlos, pero no mostrarlos, ya que la capa de presentación de datos es la vista.

Controlador: Su responsabilidad es procesar y mostrar los datos obtenidos por el Modelado. Es decir, este último trabaja de intermediario entre los otros dos, encargándose también de la lógica de negocio.

En la Figura 3.1 se muestra una imagen para entenderlo mejor:

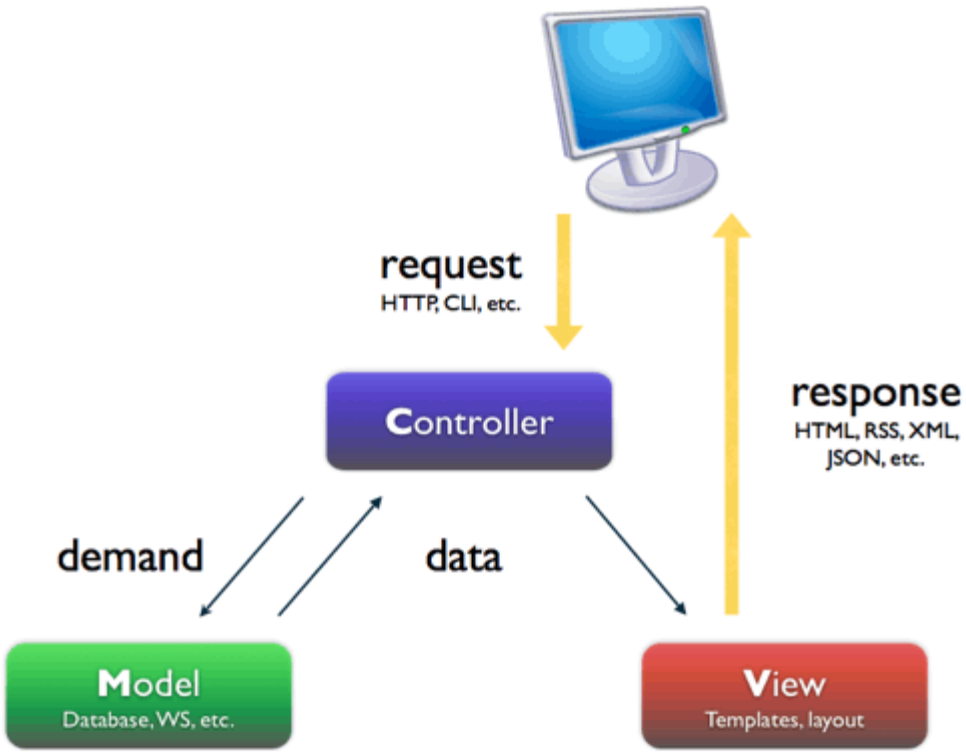

Figura 3.1 Modelo-vista-controlador

3.2 Diagrama de clases de diseño (CU Significativos)

Los diagramas de clases muestran un conjunto de clases y sus relaciones. Estos proporcionan una perspectiva estática del sistema (Booch, Rumbaugh, Jacobson, Martínez, & Molina, 1999).

En la Figura 3.2 se muestra el diagrama de clases del diseño

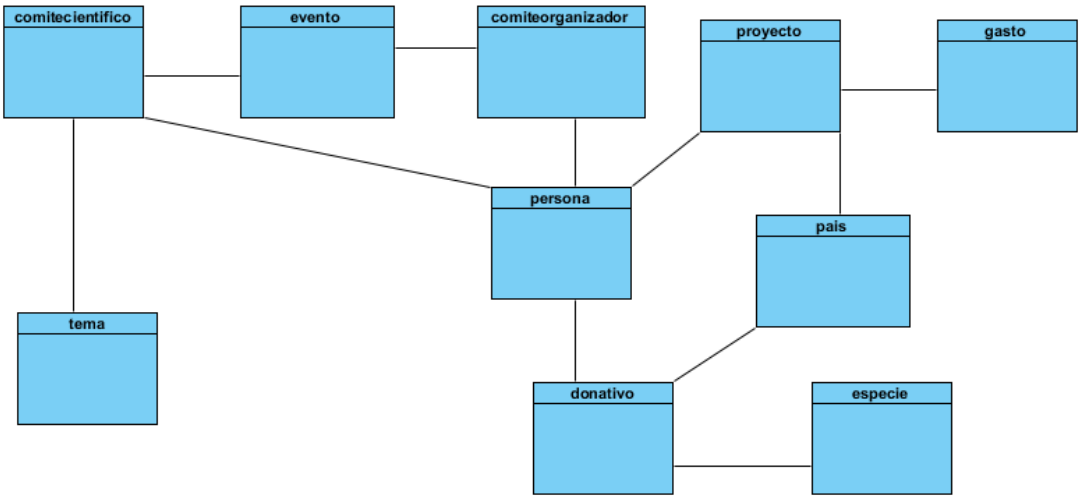

Figura 3.2 Diagrama de clases del Diseño

3.3 Diagrama de secuencia (CU Significativos)

Un diagrama de secuencias modela la interacción entre los objetos de un sistema de software (Booch, 2005). Mientras que el diagrama de casos de uso permite el modelado de una vista del escenario del negocio, el diagrama de secuencia contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario, y mensajes intercambiados entre los objetos.

3.3.1 Caso de Uso del Sistema: Insertar Proyecto

En la Figura 3.3 se muestra el Diagrama de Secuencia correspondiente al caso de uso Insertar Proyecto.

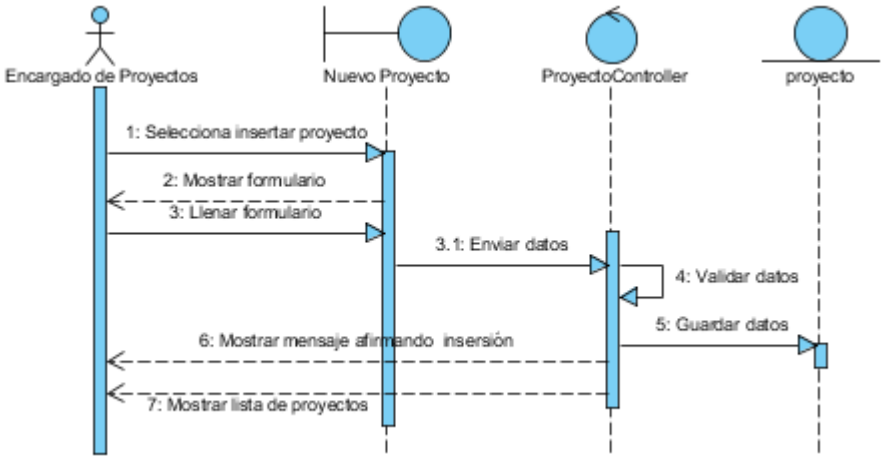

Figura 3.3 Diagrama de Secuencia Caso de Uso Insertar Proyecto

3.3.2 Caso de Uso del Sistema: Insertar Evento

En la Figura 3.4 se muestra el Diagrama de Secuencia correspondiente al caso de uso Insertar Evento.

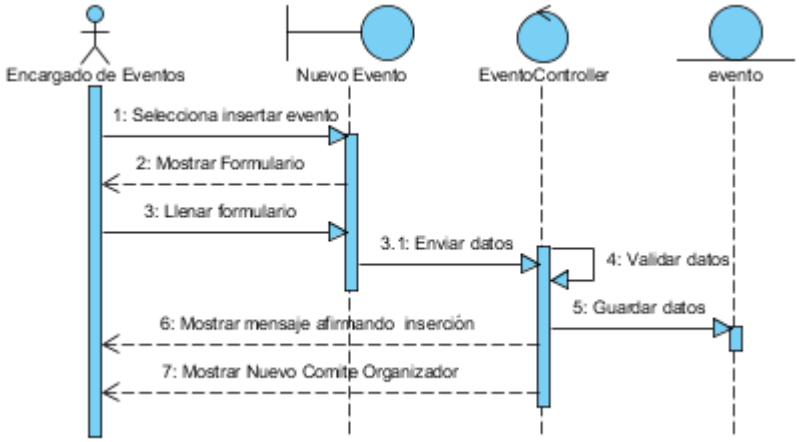

Figura 3.4 Diagrama de Secuencia Caso de Uso Insertar Evento

3.3.3 Caso de Uso del Sistema: Insertar Donativo

En la Figura 3.5 se muestra el Diagrama de Secuencia correspondiente al caso de uso Insertar Donativo.

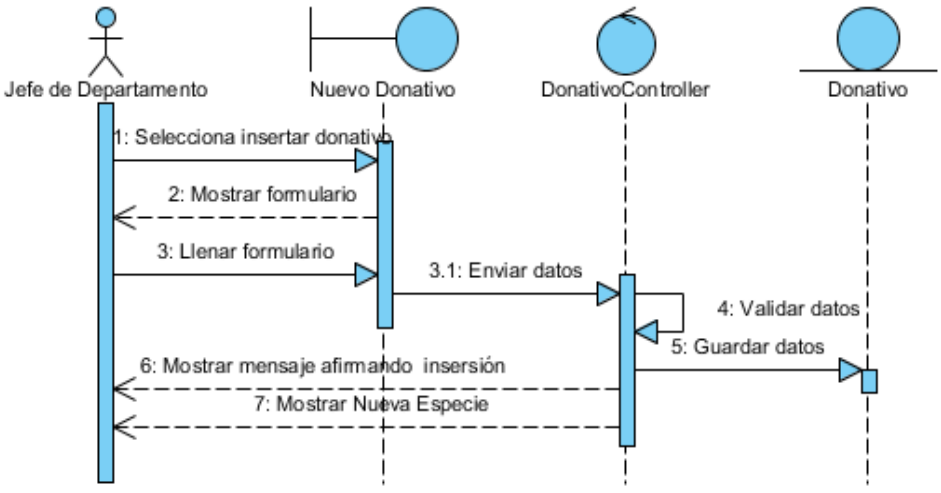

Figura 3.5 Diagrama de Secuencia Caso de Uso Insertar Donativo

3.4 Tratamiento de errores

El tratamiento de es muy importante en el desarrollo de cualquier sistema ya que mediante el mismo se asegura que los datos a insertar en la base de datos sean

correctos ya sea informándole que dejo algún campo vacío o que algún dato que introdujo es incorrecto.

Para la validación de los campos del formulario se usaron los contrains que ofrece Symfony mediante el prefijo "@Assert\ "como por ejemplo NotBlank (), que comprueba que el valor introducido no sea nulo o una cadena de texto vacía.

También se validaron los campos que no podían contener números y debían empezar con mayúscula con el contrain Regex () al cual se le pasa una expresión regular y un mensaje a mostrar en caso de que no se cumpla

Ej.: @Assert\Regex (pattern:'/^ [A-Z]+ ([a-z A-ZáéíóúÁÉÍÓÚñÑ-])*$/', match=false, message="Datos incorrectos")

Además se validó el campo correo usando la contrain Email () la cual verifica que el correo insertado tenga la estructura correcta.

3.5 Diseño de la base de datos

Una base de datos correctamente diseñada permite obtener acceso a información exacta y actualizada. Puesto que un diseño correcto es esencial para lograr los objetivos fijados para la base de datos, parece lógico emplear el tiempo que sea necesario en aprender los principios de un buen diseño ya que, en ese caso, es mucho más probable que la base de datos termine adaptándose a sus necesidades y pueda modificarse fácilmente.

3.5.1 Modelo conceptual de datos

El modelo conceptual de base de datos describe la estructura de toda la base de datos, este oculta los detalles de las estructuras de almacenamiento físico y se concentra en describir las entidades y sus relaciones (Elmasri, Navathe, & Díaz, 2007).

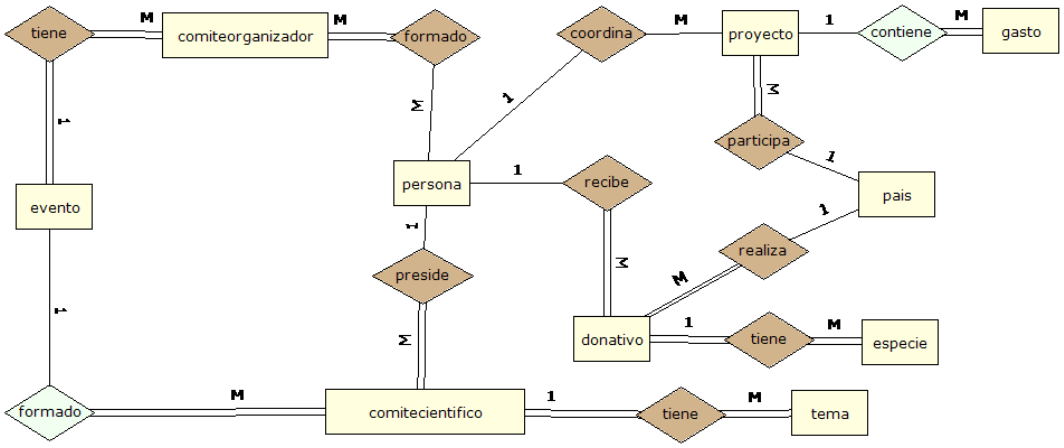

Figura 3.6 Modelo conceptual de datos

3.5.2 Modelo físico de datos

El modelo físico de datos describe cómo se almacenan realmente los datos. En este se escriben en detalle las estructuras de datos complejas de bajo nivel (Korth, Silberschatz, Sudarshan, & Pérez, 1993).

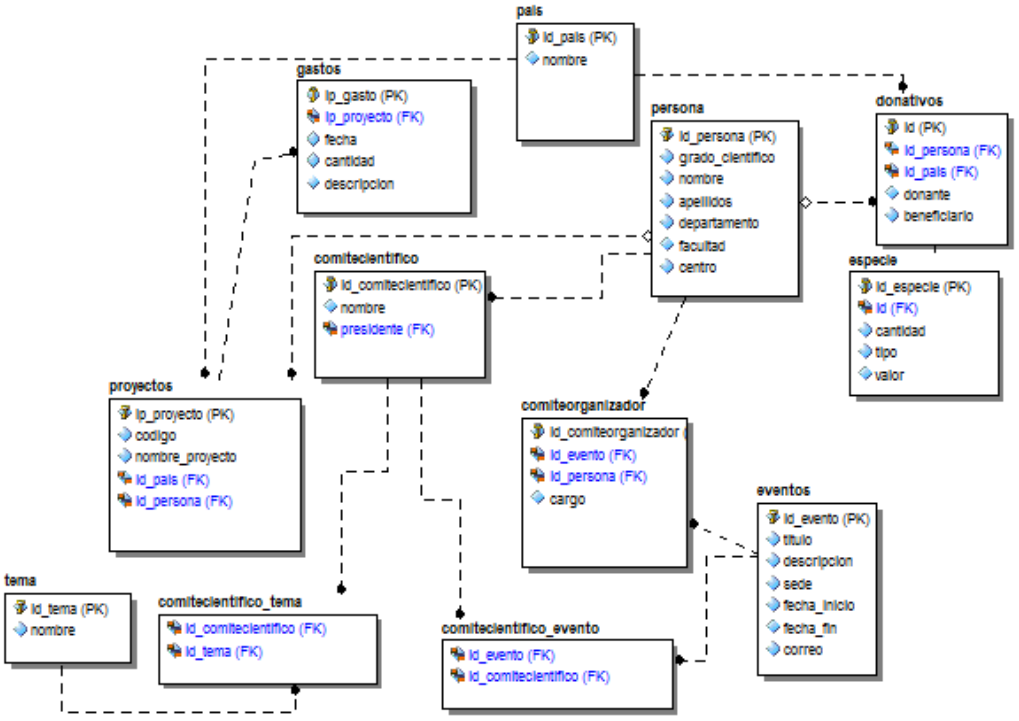

Figura 3.7 Modelo físico de datos

3.6 Modelo de componentes

El modelo de componentes permite ver el modelado de un sistema o subsistema. Muestra la organización y las dependencias lógicas entre un conjunto de componentes software como son: componentes de código fuente o componentes de código ejecutable.(Grau & Segura, 2001)

En la Figura 3.8 se muestra el modelo de componentes correspondiente a la propuesta de solución, mostrando un componente de tipo file llamado principal, este nos conduce a la vista, mediante la vista se accede al controlador el cuál es el encargado de gestionar toda la lógica del software para su correcto funcionamiento, además del controlador la vista está relacionada con varias bibliotecas, tales como la biblioteca Boostraps, la cual actúa como complementario de la vista, la biblioteca jQuery, brindado todos los métodos JavaScripts necesarios para el funcionamiento correcto de la vista, también se encuentra la DataTable la cual posee las funciones correspondientes para el filtrado de los datos en las tablas.

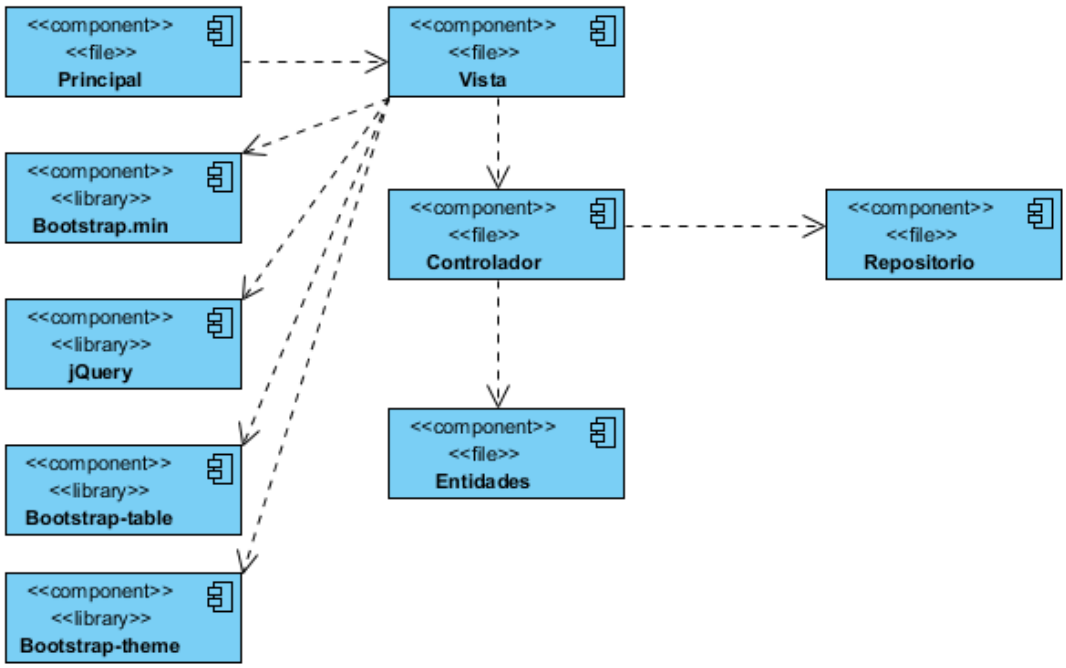

Figura 3.8 Modelo de Componentes

3.7 Diagrama de despliegue

Los Diagramas de Despliegue muestran la disposición física de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos (Torres, 2004).

En la Figura 3.9 se muestra el diagrama de despliegue correspondiente a la propuesta de solución donde el primer nodo representa la máquina del cliente el cual se conecta usando el componente navegador mediante el protocolo http. El Servidor DPE provee los servicios del sistema a los nodos clientes, almacena todos los archivos que conforman el entorno y procesa las solicitudes de los clientes en la red, este a su vez se conecta vía TCP/IP al nodo servidor de base de datos en el cual se encuentra montado el servidor de PostgreSQL

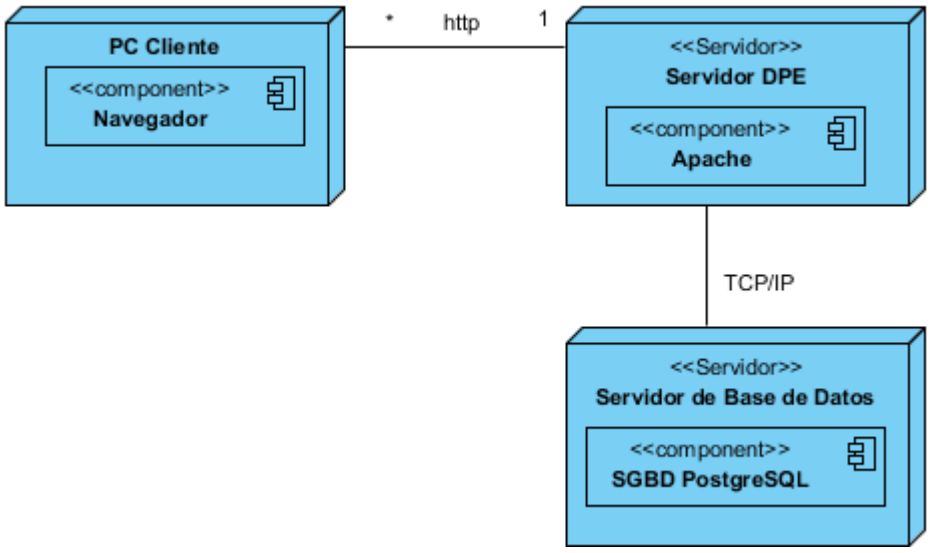

Figura 3.9 Diagrama de despliegue

3.8 Conclusiones Parciales

A lo largo del Capítulo 3 se abordó sobre la propuesta de solución al problema de investigación a través de los distintos diagramas que ayudan a comprender mejor el funcionamiento interno del sistema.

Capítulo 4 Pruebas y análisis de factibilidad

En el presente capítulo se desarrolla además de los casos de prueba apoyados en caja negra, el análisis de factibilidad utilizando la planificación basada en uno de los métodos de estimación, siendo el de casos de uso el seleccionado para esta ocasión.

4.1 Planificación basada en uno de los métodos de estimación

La planificación es un proceso de toma de decisiones para alcanzar un futuro deseado, teniendo en cuenta la situación actual y los factores internos y externos que pueden influir en el logro de los objetivos.

4.1.1 Estimación basada en Casos de Uso

La estimación mediante el análisis de puntos de casos de uso es un método propuesto originalmente por Gustav Karner de Objectory AB, y posteriormente refinado por muchos otros autores. Se trata de un método de estimación del tiempo de desarrollo de un proyecto mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores (Josue Carralero Iznaga, 2006).

4.1.1.1Estimación del Esfuerzo Basada en Casos de Uso.

Cálculo de Puntos de Casos de Uso sin Ajustar (UUCP).

Este valor se calcula a partir de la siguiente ecuación:

UUCP = UAW + UUCW donde,

UUCP: Puntos de Casos de uso sin ajustar

UAW: Factor de peso de los actores sin ajustar

UUCW: Factor de peso de los casos de uso sin ajustar

Determinación del factor de peso de los actores sin ajustar (UAW).

Este valor se calcula mediante un análisis de la cantidad de Actores presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los actores se establece, teniendo en cuenta en primer lugar, si se trata de una persona o de otro sistema, y en segundo lugar, la forma en que el actor interactúa con el sistema como aparece a continuación en la Tabla 4.1 en correspondiente a los factores de peso de los actores:

Tabla 4.1 Factor de peso de Actores sin Ajustar

Tipo de actor	Descripción	Factor de peso	Número de actores	*Resultado*
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación(API, Aplication Programming Interface)	1	0	*0*
Promedio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto.	2	0	*0*
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3	7	*18*
Total				*21*

De manera que **UAW** = 21

Determinación del factor de peso en los casos de uso sin ajustar (UUCW).

Este valor se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los casos de uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como una secuencia de actividades atómicas, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia. En la Tabla 4.2se muestran los factores de peso correspondiente a los casos de uso.

Tabla 4.2 Factores de peso de los casos de uso.

Tipo de caso de uso	Descripción	Factor de	Número de Casos de Uso	*Resultado*

		peso		
Simple	1-3 Transacciones	5	2	*10*
Promedio	4-7 Transacciones	10	9	*90*
Complejo	Mayor de 8 Transacciones.	15	0	*0*
		Total		*100*

UUCW = 100

Calculando

UUCP = UAW + UUCW

UUCP = 21 + 100

UUCP = 121

Cálculo de Puntos de Casos de Uso ajustados.

Seguidamente de calcular los Puntos de Casos de Uso sin ajustar, se debe ajustar este valor mediante la siguiente ecuación:

UCP = **UUCP** x **TCF** x **EF** donde,

UCP: Puntos de Casos de Uso ajustados

UUCP: Puntos de Casos de Uso sin ajustar

TCF: Factor de complejidad técnica

EF: Factor de ambiente

Determinación del factor de complejidad técnica (TCF).

Este coeficiente se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5, donde 0 significa un aporte irrelevante y 5 un aporte muy importante. En la Tabla 4.3 se muestran los factores de complejidad técnica a tener en cuenta.

Tabla 4.3 Factores de complejidad técnica.

Número de factor	Descripción	Peso	Valor	Factor	*Comentario*
T1	Sistema Distribuido	2	2	4	*El sistema es Web, por lo que posee cierto nivel de distribución*
T2	Tiempo de respuesta	1	1	1	*El tiempo de respuesta del sistema es mínimo.*
T3	Eficiencia por el usuario	1	1	1	*Los usuarios no tienen que ser eficientes.*
T4	Proceso interno complejo	1	0	0	*No existe un procesamiento complejo, no se ejecutan operaciones complejas.*
T5	Reusabilidad	1	3	3	*Se desea que el código sea lo más reutilizable posible por las magnitudes que puede alcanzar el software.*
T6	Facilidad de instalación	0.5	0	0	*No tiene ninguna dificultad de instalación.*
T7	Facilidad de uso	0.5	4	2	*El software no debe tener una dificultad de uso mínima.*
T8	Portabilidad	2	0	0	*Los usuarios del sistema no pretenden hacer cambios en el SO.*
T9	Facilidad de	1	3	3	*El sistema está estructurado para que los cambios no afecten*

	cambio				*su funcionalidad.*
T10	Concurrencia	1	2	2	*La concurrencia es tratada con suma importancia pues pueden existir varios clientes conectados a la vez en el mismo instante.*
T11	Objetivos especiales de seguridad	1	3	3	*El sistema solo te permite realizar las funcionalidades asociadas a tu rol.*
T12	Acceso directo a terceras partes	1	3	3	*A la aplicación puede acceder cualquier usuario.*
T13	Facilidades especiales de entrenamiento a usuarios finales	1	0	0	*No es necesario el entrenamiento de los usuarios finales, ya que se incluye un manual de usuario para garantizar la correcta usabilidad de dicho sistema.*
Total Factor				*22*	

El Factor de complejidad técnica se calcula mediante la siguiente ecuación:

TCF = 0.6 + 0.01 * Σ (Pesoix Valor asignadoi)

TCF = 0.6 + 0.01* 22

TCF = 0.82

Determinación del factor ambiente (EF).

Las habilidades y el entrenamiento del grupo involucrado en el desarrollo tienen un gran impacto en las estimaciones de tiempo. Los factores se muestran en el cálculo del factor de ambiente que se muestra en la Tabla 4.4.

Tabla 4.4 Factores de ambiente.

Número del factor	Descripción	Peso	Valor	Factor	*Comentario*
E1	Familiaridad con el modelo del proyecto usado.	1.5	4	6	*Se está familiarizado con el modelo del proyecto.*
E2	Experiencia en la aplicación	0.5	5	2.5	*Se ha adquirido suficiente experiencia durante el período de trabajo con el estudio lenguaje a utilizar.*
E3	Experiencia OO.	1	5	5	*Se considera un alto grado de experiencia en la programación orientada a objetos (OO), debido a que esta es la que se ha estudiado en cursos anteriores.*
E4	Capacidad del analista líder.	0.5	2	1	*No existe analista líder.*
E5	Motivación.	1	5	5	*Existe alta motivación ya que con este proyecto se va a agilizar la ejecución de los procesos del DPE.*
E6	Estabilidad de los requerimientos.	2	4	8	*Aunque el sistema se encuentra sujeto a cambios, este brinda las funcionalidades esenciales que dan cumplimiento a los objetivos*

					que iniciaron su realización.
E7	Personal media jornada.	-1	0	0	*No existe personal de media jornada.*
E8	Dificultad en lenguaje de programación.	-1	0	0	*Se considera una dificultad media del lenguaje empleado ya que este ofrece grandes facilidades y ventajas.*
Total				*27.5*	

El factor de ambiente se calcula mediante la siguiente ecuación:

EF = 1.4 – 0.03 * Σ (Pesoix Valor asignadoi)

EF = 1.4 – 0.03 * 27.5

EF = 0.58

Cálculo de los Puntos de Casos de Uso Ajustados:

UCP = UUCP * TCF * EF

UCP = 121 * 0.82* 0.58

UCP = 57.55

Cálculo del esfuerzo.

El esfuerzo en horas-hombre viene dado por:

E = **UCP** * **CF** donde:

E: esfuerzo estimado en horas-hombre.

UCP: Puntos de casos de uso ajustados.

CF: Factor de conversión (20 horas-hombre por defecto).

E= 57.55* 20

E = 1151 Horas-Hombre

Para la obtención de una estimación más exacta de la duración del proyecto, se hace necesario agregar a la estimación del esfuerzo obtenida por los puntos de casos de uso, las estimaciones de esfuerzo de las restantes actividades que se llevaron a cabo durante el desarrollo del software; así la distribución del esfuerzo entre dichas actividades está dada por la siguiente aproximación mostrada en la Tabla 4.5.

Tabla 4.5 Distribución genérica del esfuerzo

Actividad	*Porcentaje*
Análisis	*10.00%*
Diseño	*10.00%*
Programación	*70.00%*
Pruebas	*5.00%*
Sobrecarga(otras actividades)	*5.00%*

Con este criterio y tomando como entrada la estimación de tiempo calculada a partir de los puntos de casos de uso, se pueden calcular las demás estimaciones para obtener la duración total del proyecto. En la Tabla 4.6Tabla **4.6** se muestra la distribución real del esfuerzo.

Tabla 4.6 Distribución real del esfuerzo.

Actividad	*Porcentaje*
Análisis	164
Diseño	164
Programación	1151
Pruebas	82

Sobrecarga(otras actividades)	82
Total	1644

Cálculo del esfuerzo total:

ETotal = *1644* horas / hombre

Cálculo del tiempo de desarrollo:

TDesarrollo = ETotal/CHTotalCHTotal: Cantidad de hombres =1

TDesarrollo = *1644*/1 horas

TDesarrollo =1644 horas

Considerando que se trabajan 8 horas diarias:

TDesarrollo = TDesarrollo/8 horas/día

TDesarrollo= 1644 horas/8 horas/día

TDesarrollo= 205 días aproximadamente

Cálculo del costo:

Costo Total = ETotal * 1 * TH

TH: El salario promedio de 1 desarrollador es de $600 y por tanto la **TH** = 600 / 160 = 3.75

Costo Total = 1644* 1 * 3.75

Costo Total = $6165

Conclusiones del Análisis de Factibilidad

Teniendo en cuenta los resultados obtenidos correspondientes a la factibilidad del software se apoya la realización del sistema, ya que es factible y económico. El mismo implicará un esfuerzo total de 1644 horas /hombre, para un tiempo de desarrollo de 205 días aproximadamente, se contarán con 1 hombre para su realización, lo que implica un costo de $6165 para una tarifa horaria de $3.75.

4.2 Casos de Pruebas (Caja negra)

Se denomina Caja Negra a aquel elemento que es estudiado desde el punto de vista de las entradas que recibe y las salidas o respuestas que produce, sin tener en cuenta su funcionamiento interno. En otras palabras, de una *caja negra* nos interesará su forma de interactuar con el medio que le rodea (en ocasiones, otros elementos que también podrían ser *cajas negras*) entendiendo qué es lo que hace, pero sin dar importancia a cómo lo hace(Roger S Pressman & Troya, 1988).

Según (R. PRESSMAN, 2002), las pruebas de caja negra, también denominada prueba de comportamiento, se centran en los requisitos funcionales del software. Estas pruebas permiten obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa.

A continuación se muestran las pruebas de caja negra correspondientes a los casos de uso Insertar Evento en la Tabla 4.7 e Insertar Donativo en la Tabla 4.8.

Tabla 4.7 Insertar Evento

Campo	Entrada 1	Entrada 2	Entrada 3	Entrada 4 (Válida)
Título	-	Convención de master	Convención de master	Convención de master
Email	-	yrtoledo	yrtoledo@uclv.cu	yrtoledo@uclv.cu
Duración	-	-	f	5

Resultado esperado

Entrada 1: El sistema muestra un error diciendo que el título no puede estar vacío.

Entrada 2: El sistema muestra un error informando que entre un email correcto.

Entrada 3: El sistema nos informa que se guardaron los datos y nos re-direcciona a la página de Nuevo Comité Organizador.

Tabla 4.8 Insertar donativo

Campo	Entrada 1	Entrada 2 (Válida)
Donante	-	Marco Polo
Receptor	Pedro León	Pedro León
fecha	05/05/2016	05/05/2016
País Donante	Alemania	Alemania

Nota: En el caso del Receptor, la fecha y el País Donante el sistema siempre pone un valor por defecto.

Resultado esperado

Entrada 1: El sistema muestra un error diciendo que el donante no puede estar vacío.

Entrada 2: El sistema nos informa que se guardaron los datos y nos re-direcciona a la página de Nuevo elemento del donativo.

Conclusiones parciales

En este capítulo, se efectuó un análisis de factibilidad utilizando el método de estimación basada en casos de uso lo que permitió obtener un estimado del esfuerzo, el tiempo y el costo que figura llevar a cabo el desarrollo de un proyecto; además se efectuaron pruebas de caja negra efectuadas sobre la interfaz del software con el fin de demostrar que las funciones del software son operativas

Conclusiones

Como resultado de esta investigación se desarrolló un sistema para la gestión de la información del DPE, siendo este sistema de gran utilidad para los trabajadores de dicho departamento y cumpliendo de esta forma con los objetivos planteados:

1. Se determinaron los requisitos a informatizar realizando un análisis profundo del departamento.
2. Se realizó un diseño de base de datos que permitió almacenar los datos del departamento.
3. Se diseñó un sistema utilizando las tecnologías web para gestionar la información del departamento.
4. Se evaluó la validez del sistema mediante las pruebas de caja negra realizadas al mismo.

Recomendaciones

Luego de haber concluido la investigación quedan algunas recomendaciones a considerar para un futuro perfeccionamiento de la aplicación:

1. Adaptar el software a los usuarios de la base datos de la UCLV.
2. Agregar un módulo para la gestión de los contratos.

Referencias Bibliográficas

Adell, J., & Bernabé, Y. (2007). Software libre en educación. *Tecnología educativa. Madrid: McGraw-Hill*, 173-195.

Ardissone, J. (2012). Desarrollo de aplicaciones web con Symfony. Introduccion a Symfony 2.

Armand, S. (2014). *Extending Symfony2 Web Application Framework*: Packt Publishing Ltd.

Banker, K. (2011). *MongoDB in action*: Manning Publications Co.

Booch, G. (2005). *The unified modeling language user guide*: Pearson Education India.

Booch, G., Rumbaugh, J., Jacobson, I., Martínez, J. S., & Molina, J. J. G. (1999). *El lenguaje unificado de modelado* (Vol. 1): Addison-Wesley.

Bykbaev, V. R. (2008). Lenguajes de Scripting:¿ una nueva forma de programar? *Ingenius*(2).

Cochran, D. (2012). Twitter Bootstrap Web Development. *(1st edición)*.

Chaffer, J., & Swedberg, K. (2007). *Learning jquery: better interaction design and web development with simple javascript techniques*: Packt Publishing.

Chavarría, J. V. (2011). Software libre, alternativa tecnológica para la educación. *Revista Actualidades Investigativas en Educación, 5*(2).

DEGLOVANNINI, M. (2005). Comparativa de Frameworks WEB

Dondo, A. (2006). ¿ Por qué elegir PHP.

Duque, R. G. (2011). Python para todos.

Eguiluz, J. (2013). Desarrollo web ágil con Symfony2. *España: Gestor de publicaciones*.

Elmasri, R., Navathe, S. B., & Díaz, J. M. (2007). Fundamentos de sistemas de bases de datos.

Fabien Potencier, R. W. (2013). *Symfony 2.4, el libro oficial*.

Features of Yii. (2016). Retrieved 22 de marzo de 2016., from http://www.yiiframework.com/features/

Free Software Foundation, I. (2016). ¿Qué es el software libre? , Consultado 24/5/2016, from http://www.gnu.org/philosophy/free-sw.es.html

Frittelli, V., Serrano, D., Teicher, R., Steffolani, F., Tartabini, M., Fernández, J., & Bett, G. (2013). Uso de Python como Lenguaje Inicial en Asignaturas de Programación. *Editor Responsable*, 132.

Gallego, J. A. (2003). Desarrollo Web con PHP y MySQL

Grau, X. F., & Segura, M. I. S. (2001). Desarrollo orientado a objetos con UML. *Facultad de Informática–UPM, Articulo, 3*.

Group, T. P. G. D. (2013). PostgreSQL 9.3.0 Documentation.

Gutiérrez, J. D. (2004). *Desarrollo web con PHP 5 y MySQL*.

Gutiérrez, J. J. (2014). ¿ Qué es un framework Web? *Available in: http://www. lsi. us. es/~ javierj/investigacion_ficheros/Framework. pdf Accessed May, 12*.

Holovaty, A., & Kaplan-Moss, J. (2009). *The definitive guide to Django: Web development done right*: Apress.

Ivar Jacobson, G. B., JAMES Rh (2000). El Proceso Unificado de Desarrollo de SW.

Josue Carralero Iznaga, J. F. T. A., Julio Martínez Prieto, José A. Franco Navarro, Saylis Cabrera Sierra, Paula Ardanza Menéndez, Norailis Brito Hernández. (2006). ¨Hotel X¨ Un ejemplo práctico para la asignatura de Ingeniería de Sosfware 1.

Korth, H. F., Silberschatz, A., Sudarshan, S., & Pérez, F. S. (1993). *Fundamentos de bases de datos*: McGraw-Hill.

LIBRE, S. (2010). Diez ventajas del software libre y propietario.

Mariño, J. C. G. (2006). B-Learning utilizando software libre, una alternativa viable en Educación Superior. *Revista complutense de Educación, 17*(1), 121.

Martínez Echeverry, M. A. (2015). Página web sala situacional Risaralda.

Marzal, A., & Gracia, I. (2003). Introducción a la programación con Python. *Universitat Jaume I*.
MySQL, A. (2000). Mysql database.
MySQL, A., & MySQL, A. (2008). The world's most popular open source database. *MySQL AB*.
Pacheco, N. (2011). Doctrine 2 ORM Documentation
Potencier, F., & Zaninotto, F. (2008). Symfony, la guía definitiva. *Libros Web.[Online] www. librosweb. es*.
PRESSMAN, R. (2002). INGENIERIA DEL SOFTWARE UN ENFOQUE PRACTICO.
Pressman, R. S. (2010). Software Engineering : A practitioner´s Approach.
Pressman, R. S., & Troya, J. M. (1988). *Ingeniería del software*: McGraw Hill.
Ramırez, A. O. (2010). Python como primer lenguaje de programación. *Publicación interna del Tecnológico de Monterrey, Campus Estado de México*.
Russell, C. (2016). Dependency Management *PHP Development Tool Essentials* (pp. 67-82): Springer.
Silberschatz, A., Korth, H. F., Sudarshan, S., Pérez, F. S., Cordero, A. G., & Fernández, J. C. (2002). *Fundamentos de bases de datos*: McGraw-Hill.
Stallman, R. (2004). *Software libre para una sociedad libre*: Madrid: Traficantes de Sueños, 2004.
Stonebraker, M., & Kemnitz, G. (1991). The POSTGRES next generation database management system. *Communications of the ACM, 34*(10), 78-92.
Torres, P. L. (2004). Desarrollo de Software Orientado a Objeto usando UML. *Universidad Politecnica de Valencia (UPV)–España*.
Tupe, C., & Cisneros, J. (2008). Evaluación y Selección de Framework de Desarrollo PHP: Symfony, Kumbia, CakePHP y Zend.
Ullman, L. (2013). The Yii Book: Developing Web Applications Using the Yii P HP Framework: Selfpublished.
Vaswani, V. (2010). *Zend Framework, A Beginner's Guide*: McGraw Hill Professional.
Zend framework About. (2016). 2016, from https://framework.zend.com/about
Zikopolous, P. C., Baklarz, G., & Scott, D. (2005). *Apache Derby/IBM Cloudscape*: Prentice Hall PTR.