Universidad de las Ciencias Informáticas
Facultad 15

Título: Implementación del Módulo Gestión de Actividades del Tutor Virtual de Evaluación para el Aprendizaje Autónomo de Idiomas

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas.

Autores: Alberto Elers Pérez
Jorge Luis Sariol Pérez

Tutores: MSc. Yoan Martínez Márquez
Ing. Reinier Castillo González
Lic. Moisés Alain Mayet Solano

Ciudad de La Habana, Junio de 2010

Año 52 de la Revolución

DECLARACIÓN DE AUTORÍA

Declaramos ser los autores del presente trabajo de diploma y autorizamos a la Universidad de las Ciencias Informáticas para que hagan uso pertinente de este trabajo.

Para que así conste firmamos el presente a los __ días del mes de _____.

_____ _____

Firma de Autor Firma de Autor

_____ _____

Firma Tutor Firma Tutor

_____

Firma Tutor

AGRADECIMIENTOS

A los que tuvieron que ver en mi formación durante todo este tiempo. A mis profesores, por haberme enseñado y soportado.

A mis tutores por estar siempre cuando los necesitaba y por su gran apoyo, sin ellos este trabajo no se hubiera realizado.

A mi familia, por ayudarme tanto y por hacerme ver que si se puede.

A Senovia (mi Madre), a Sariol (mi Padre), a FIDE (mi Hermano), por su apoyo y por creer siempre en mi; Los quiero.

A mi compañero Alberto, por haberme ayudado tanto con la tesis.

A mi primo Walfro, por estar siempre conmigo, este es tu segundo título.

A mis amigos y amigas, los que vienen desde 1ro y a los de más adelante.

A todos en general.

S@riol

A mis padres por haberme educado y guiado por buen camino, por darme todo su apoyo y amor incondicional.

A mi hermana por ser la mejor hermana del mundo, por estar ahí para mí siempre, en todo momento.

A mi abuela por sus consejos sabios.

A mis tías por ser mis segundas madres, por estar siempre en cada momento que hizo falta.

A mis primas quienes han sido más que primas mis hermanas.

A mis tutores por su apoyo y confianza depositada en mí.

A mi compañero de tesis por su amistad y por poner todo su esfuerzo en este trabajo.

A todos mis amigos del proyecto VIRTEVALL.

A mis amigos y compañeros del aula, por brindarme su amistad desinteresada y compartir momentos buenos y malos.

Alberto

DEDICATORIA

A toda mi familia, en especial a mi HERMANO y a mis padres.
A Walfro; como dije antes, ya tienes dos títulos.
A todos aquellos, que han pensado en rendirse, vean que solo hay que esforzarse un poco.
A todo: el que creyó en mí, el que en algún momento me brindó su ayuda.
A mis amigos y amigas.
A todos.

S@riol

Dedico este trabajo a toda mi familia por estar presentes en todo momentos, por su dedicación, por su confianza, por su apoyo y por su amor incondicional que me han brindado durante toda mi vida.

Alberto

RESUMEN

El proceso de formación a lo largo de los años ha sufrido cambios con el objetivo de mejorar muchos aspectos que son considerados de gran importancia en el proceso de aprendizaje. La inclusión de las potencialidades de las Tecnologías de la Información y las Comunicaciones (TICs) ha dado paso al surgimiento de nuevos métodos como el autoaprendizaje, para lo cual han surgido recursos como los espacios virtuales, aulas virtuales, tutores virtuales, entre otros.

La UCI es un centro que está a la vanguardia de todo el proceso de enseñanza y autoaprendizaje es por ello que surge el Proyecto de Innovación Pedagógica: Metodología de Evaluación para el Aprendizaje Autónomo de Idiomas (VIRTEVALL), con el cual se prevé crear un Tutor Virtual de Evaluación para el Aprendizaje Autónomo de Idiomas que funcione como un ambiente alternativo de evaluación para aprendizaje autónomo de Idiomas, que le permite a los estudiantes trabajar de manera independiente en sus debilidades y profundizar en los temas que son de su interés, a través de actividades evaluativas personalizadas.

La presente investigación persigue como objetivo implementar el módulo gestión de actividades del *Tutor Virtual de Evaluación para el Aprendizaje Autónomo de Idiomas* en la UCI, para lo cual es usado Symfony 1.2.8 como framework de desarrollo, como lenguaje de programación empleado PHP 5.2.5 y como IDE NetBeans 6.8 M2. Además como SGBD se utilizó PostgreSQL- 8.3.4-1.

Para la construcción de la aplicación fue necesario realizar una valoración crítica del diseño, sobre lo propuesto por los analistas, teniendo en cuanta los requisitos de la aplicación, se hizo un estudio acerca de los componentes a reutilizar y los algoritmos no triviales a implementar, para finalmente describir las nuevas clases y operaciones necesarias para darle solución al objetivo trazado en la presente investigación. Para la validación, se aplicaron técnicas de prueba de software que permitieron comprobar la validez de la solución propuesta. Se detallaron los casos de prueba con sus objetivos y alcance, que permitieron detectar la ocurrencia de errores en la aplicación y la posterior solución de los mismos.

PALABRAS CLAVES

“Tutor Inteligente, Implementación, Actividades Evaluativas, autonomía.”

INDICE

INDICE DE FIGURAS:

INDICE DE TABLAS:

INTRODUCCION

Aprender y enseñar es una necesidad de la humanidad, donde su desarrollo ha posibilitado que el Proceso de Enseñanza Aprendizaje (PEA) evolucione para lograr una mayor calidad. El empleo de las tecnologías de la información y las comunicaciones han contribuido en gran medida a impulsar este proceso, donde cada vez más el estudiante desarrolla su aprendizaje de forma autónoma. Estas tecnologías han abierto una nueva ventana en el proceso enseñanza-aprendizaje, ya que permite disponer de recursos altamente orientados a la interacción y el intercambio, favoreciendo la integración de los conocimientos de los estudiantes.

Hoy en día la pedagogía cubana se encuentra inmersa en cambios revolucionarios y transformadores en todos los niveles de enseñanza. Dentro de ella, el desarrollo del PEA está sufriendo cambios en aras de lograr la formación de un individuo pleno, poseedor de una cultura general integral, que le permita ser protagonista activo en un mundo donde los cambios en el campo de la información y la tecnología requieren cada vez de mayor preparación. Estos tutores considerados "*Sistemas Tutor Inteligente* (STI)", tienen como propósito exhibir un comportamiento parecido al de un tutor humano, por tal razón, tienen que ser capaces de adaptarse al comportamiento mostrado por un estudiante. El uso de los STI está muy difundido por todo el mundo, sin embargo, en Cuba sólo se tiene conocimiento de uno de estos sistemas, tal es el caso del *Sistema Tutorial Inteligente para Diagnóstico y Tratamiento de Infecciones de Transmisión Sexual (STIITS)*, lo que demuestra la poca difusión de este tipo de medios educativos en el país.

En la Universidad de las Ciencias Informáticas (UCI), institución que no está exenta en la informatización en el proceso educativo, el primer paso fue la creación del Proyecto de Innovación Pedagógica *Centro Virtual de Autoaprendizaje de Lenguas Extranjeras (CEVALE)*, que surgió a partir del re-diseño de la estrategia curricular de la disciplina de Idioma Inglés, el cual no pudo ser terminado por múltiples razones que no son objetos de esta investigación.

En el año 2009, se crea el Proyecto de Innovación Pedagógica: Metodología de evaluación para aprendizaje autónomo de Idiomas Extranjeros (VIRTEVALL). Con la puesta en marcha de este proyecto se prevé crear un ambiente alternativo de evaluación para aprendizaje autónomo de Idiomas Extranjeros,

que favorezca el trabajo de los estudiantes de manera independiente en sus debilidades y profundizar en temas que sean de su interés en el aprendizaje autónomo de inglés.

El Proyecto de Innovación Pedagógica está estructurado sobre la base del sustento teórico provisto por una tesis doctoral sobre evaluación para aprendizaje autónomo de Idiomas Extranjeros. Como aporte práctico, ofrece el STI para implementar la metodología elaborada, el mismo está sustentado en el desarrollo de dos tesis de maestría que abordarán la integración de las tecnologías en la gestión de la evaluación para aprendizaje autónomo de Idiomas Extranjeros y la integración de algoritmos de inteligencia artificial en la gestión de la evaluación para aprendizaje autónomo de Idiomas Extranjeros.

En una primera etapa, por la complejidad del sistema a desarrollar, se definió por la dirección del proyecto la división en nueve módulos estructurados en dependencia de las diferentes funcionalidades identificadas, entre ellos se encuentra el de Actividades, el cual tiene como su principal objetivo gestionar las actividades evaluativas de entrenamiento para aprendizaje autónomo de Idiomas Extranjeros de los estudiantes matriculados en el sistema.

Este STI actualmente no cuenta con un módulo de actividades que permita a los estudiantes, entrenarse en las habilidades y los conocimientos que vayan adquiriendo. Por lo que, se hace necesario precisar el siguiente **problema a resolver**: ¿Cómo mejorar los procesos de gestión de actividades del Tutor Virtual de Evaluación para el Aprendizaje Autónomo de Idiomas (VIRTEVALL)?

Siendo el **objeto de estudio**: los procesos de gestión de actividades para evaluar el aprendizaje autónomo de idiomas extranjeros.

Presentando como **campo de acción** en esta investigación**:** Automatización de los procesos de gestión de actividades para el *Tutor Virtual de Evaluación para el Aprendizaje Autónomo de Idiomas* en la Universidad de las Ciencias Informáticas.

A partir del problema referenciado anteriormente se puede enunciar la siguiente **idea a defender**: si se implementa de forma correcta el módulo de gestión de actividades para la evaluación del aprendizaje autónomo de idiomas extranjeros entonces se potenciarán las funcionalidades del *Tutor Virtual de Evaluación para el Aprendizaje Autónomo de Idiomas*.

Determinándose como **objetivo general:** Implementar el módulo gestión de actividades del *Tutor Virtual de Evaluación para el Aprendizaje Autónomo de Idiomas* en la Universidad de las Ciencias Informáticas.

Para darle cumplimiento al objetivo general se ha decidido desarrollar como objetivos específicos y tareas de la investigación:

- Valorar las tendencias actuales que existen en cuanto a la gestión de actividades para evaluar el auto-aprendizaje y la determinación de estas para el desarrollo de la aplicación que se va a implementar.
  - ✓ Caracterización de las tipologías de actividades para evaluar el auto-aprendizaje de idiomas extranjeros a partir del modelo planteado en una tesis doctoral.
- Seleccionar las tendencias y tecnologías actuales más apropiadas para el desarrollo del módulo.
  - ✓ Análisis de las tecnologías Web existentes.
  - ✓ Valoración de la implementación de sistemas Web similares.
- Implementar el módulo gestión de actividades.
  - ✓ Determinación de los estándares de codificación.
  - ✓ Realización de los diagramas correspondientes a la implementación.
  - ✓ Validación de los algoritmos empleados.

Métodos científicos de investigación

Para el desarrollo de la investigación se utilizarán métodos teóricos y empíricos.

Métodos teóricos

- **Análisis histórico-lógico:** Este método permitirá realizar un análisis de los elementos que se utilizan en la implementación de sistemas Web, dígase origen y evolución de los diferentes lenguajes de programación, frameworks, Entornos de Desarrollo Integrado (en lo adelante IDE) y SGBD para determinar los más idóneos para desarrollar la aplicación.

- **Analítico - Sintético:** Este método permitirá después de realizar un análisis de las tendencias actuales en torno a la investigación para la gestión de actividades, determinar los principales métodos y algoritmos que son usados a nivel mundial y que pueden servir en la implementación del módulo.

Métodos empíricos

- **Entrevista:** Se realizaron múltiples entrevistas no formales a los clientes, líderes de proyecto, analistas en aras de obtener información referente al funcionamiento del sistema.

Posibles resultados:

- Prototipo funcional del módulo gestión de Actividades del *Tutor Virtual de Evaluación para el Aprendizaje Autónomo de Idiomas.*

Estructuración del contenido

El presente trabajo está conformado por 3 capítulos.

Capítulo 1. Fundamentación teórica: Se realiza un estudio de las tendencias tecnológicas actuales, en lo que a construcción de software se refiere. Y se justifica la razón de la selección de las herramientas para solucionar el problema planteado.

Capítulo 2. Descripción y análisis de la solución propuesta: Se describen algunos algoritmos a implementar y se analiza la complejidad de los mismos, así como las clases que modelan la solución y las principales funcionalidades de cada una de ellas. Finalmente se establecen los estándares de codificación a utilizar.

Capítulo 3. Validación de la solución propuesta: Se refiere al diseño de las pruebas de unidad que permitan validar la solución propuesta, detallando de las mismas su objetivo, alcance y tipo, al igual que los valores utilizados para la ejecución de dichas pruebas consumando un sistema robusto.

CAPITULO 1 Fundamentación Teórica

En este capítulo se realizará una comparación de los distintos lenguajes de programación posibles a emplearse en la implementación del Módulo de Gestión de Actividades para el *Tutor Virtual de Evaluación para el Aprendizaje Autónomo de Idiomas* que se propone en el presente trabajo de diploma, tales como Java y PHP. Se compararán además los frameworks para PHP tales como CodeIgniter y Symfony; los frameworks del lado del cliente ExtJS, Dojo; los entornos de desarrollo como Zend Studio y NetBeans para PHP, los SGBD como MySQL y PostgreSQL. Las tecnologías utilizadas en la actualidad, también serán objeto de comparación especificando las ventajas y desventajas, así como sus características; concluyendo con una breve explicación del por qué de la selección realizada para el desarrollo definitivo de la aplicación.

1.1. Aplicación de Escritorio vs Aplicación Web

En la ingeniería de software se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores web (HTML, JavaScript, Java, ASP, PHP, etc.) en la que se confía la ejecución al navegador. (1)

Desde el punto de vista de la arquitectura se distinguen dos lados, el primero es el cliente en donde se encuentra el usuario final utilizando la aplicación por intermedio de un navegador (Internet Explorer, Mozilla Firefox, entre otros), es aquí donde el usuario interactúa con la aplicación localizada al otro lado o servidor en donde residen realmente los datos, reglas y lógica de la aplicación.

Es importante mencionar que una página Web puede contener elementos que permiten una comunicación activa entre el usuario y la información. Esto permite que el usuario acceda a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos y acceder a gestores de BD de todo tipo.

Las aplicaciones web son populares debido a lo práctico del navegador web como cliente ligero, así como la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de

usuarios potenciales. Existen aplicaciones como los webmails, wikis, weblogs, tiendas en línea que son ejemplos bien conocidos de aplicaciones web.

Las ventajas que ofrece el desarrollo de aplicaciones web son notables, entre ellas pueden citarse:

- Instalación: la principal ventaja de un servicio web es poder acceder a él y a los datos que contiene desde cualquier sitio. Si el usuario accede al sistema es que tiene instalado todo lo necesario para ejecutar la aplicación: el navegador: acceso desde cualquier equipo.
- Compatibilidad multiplataforma**:** Las aplicaciones web tienen un camino mucho más sencillo para la compatibilidad multiplataforma que las aplicaciones de software descargables. Varias tecnologías incluyendo Java, Flash, ASP y Ajax permiten un desarrollo efectivo de programas soportando todos los sistemas operativos principales.
- Actualizaciones: No hay que actualizar nada porque tampoco ha habido que instalar nada. Es decir la administración es nula: no tiene que instalarse, no tiene que configurarse, no se tiene que hacer nada, simplemente empezar a utilizarlo. El usuario tiene todas sus preferencias/configuraciones guardadas en el servidor web y no en su ordenador.
- Consumo de recursos para terceros: la mayor parte de consumo de ciclos de procesador, memoria, etcétera, ocurren en el servidor.
- Múltiples usuarios concurrentes**:** Las aplicaciones web pueden ser utilizada por múltiples usuarios al mismo tiempo.

Entre las desventajas que se destacan en las aplicaciones web se puede citar que:

- Depende de una conexión a Internet permanente (generalmente) y una conexión promedio para una óptima navegación.
- La estabilidad de la aplicación está sujeta al número de visitas en un mismo lapso de tiempo (sobre carga del servidor).

Por aplicaciones de escritorio se entiende toda aplicación que ha sido desarrollada para ser ejecutada en una plataforma específica, ya sea Windows, Linux ó Mac. El desarrollo sobre una plataforma, normalmente, implica que la aplicación "no" pueda ser ejecutada en otras. (2)

Entre las ventajas que posee las aplicaciones de escritorios se encuentran las siguientes:

- Mayor capacidad gráfica visual.
- Menor tiempo de respuesta (aplicación más rápida).
- Mayor personalización.

A pesar de esto las aplicaciones de escritorio poseen las siguientes desventajas:

- Duplicidad de datos por la falta de unificación de los mismos.
- Diseminación de la información y lógica en muchas partes (cada computadora que la use).
- Falta de portabilidad de la aplicación a diferentes sistemas operativos.
- Traumas a la hora de realizar actualizaciones o correcciones al programa ya que las instalaciones están diseminadas.
- Dificultad para configurar cada una de las instalaciones dependiendo de las necesidades de cada usuario.

Luego de analizar ambos tipos de aplicaciones, se decidió desarrollar una aplicación web, debido a que sus características propiciarán que la solución a obtener sea de fácil acceso, de un coste de producción barato y de un fácil mantenimiento.

Después de haber decidido realizar una aplicación web, es necesario tener en cuenta el paradigma de programación a utilizar para llevar a cabo el desarrollo de la misma.

1.2. Paradigmas de Programación

Representan un enfoque particular o filosofía para la construcción del software. No existe uno mejor que otro, sino que cada uno tiene ventajas y desventajas. También hay situaciones donde un paradigma resulta más apropiado que otro. (3)

1.2.1. Paradigma por Procedimientos o Paradigma Imperativo

Es tal vez el más conocido y utilizado en el proceso de programación, donde los programas se desarrollan a través de procedimientos. Pascal, C y BASIC son tres de los lenguajes imperativos más importantes. La palabra latina imperare significa "dar instrucciones". El paradigma se inició al principio del año 1950 cuando los diseñadores reconocieron que las variables y los comandos o instrucciones de asignación constituían una simple pero útil abstracción del acceso a memoria y actualización del conjunto de instrucciones máquina. Debido a la estrecha relación con la arquitectura de la máquina, los lenguajes de programación imperativa pueden ser implementados muy eficientemente, al menos en principio.

El paradigma imperativo aún tiene cierto dominio en la actualidad. Una buena parte del software actual ha sido desarrollado y escrito en lenguajes imperativos. La gran mayoría de programadores profesionales son principalmente o exclusivamente programadores imperativos (Hay que añadir que los paradigmas de la programación concurrente y orientada al objeto son en realidad sub-paradigmas de la programación imperativa, así que sus adeptos también son programadores imperativos).

1.2.2. Programación Funcional

Se caracteriza por el uso de expresiones y funciones. Un programa dentro del paradigma funcional, es una función o un grupo de funciones compuestas por funciones más simples estableciéndose que una función puede llamar a otra o el resultado de una función puede ser usado como argumento de otra función. El lenguaje por excelencia ubicado dentro de este paradigma es el LISP.

1.2.3. Paradigma Declarativo o Paradigma de Programación Lógica

Se basa en el hecho que un programa implementa una relación antes que una correspondencia. Debido a que las relaciones son más generales que las correspondencias (identificador - dirección de memoria), la

programación lógica es potencialmente de más alto nivel que la programación funcional o la imperativa. El lenguaje más popular enmarcado dentro de este paradigma es el lenguaje PROLOG[1]. El auge del paradigma declarativo se debe a que el área de la lógica formal de las matemáticas ofrece un sencillo algoritmo de resolución de problemas adecuado para usarse en un sistema de programación declarativo de propósito general.

1.2.4. Programación Orientada a Objetos

El paradigma orientado a objetos, se basa en los conceptos de objetos y clases de objetos. Un objeto es una variable equipada con un conjunto de operaciones que le pertenecen o están definidas para ellos. El paradigma orientado a objetos actualmente es el paradigma más popular y día a día los programadores, estudiantes y profesionales tratan de tomar algún curso que tenga que ver con este paradigma, podría decirse, que programar orientado a objetos está de moda.

Alrededor de 1970 David Parnas planteó el ocultamiento de la información como una solución al problema de gerenciar grandes proyectos software. Su idea fue encapsular cada variable global en un módulo con un grupo de operaciones (al igual que los procedimientos y las funciones) que permitan tener un acceso directo a la variable. Otros módulos pueden acceder a la variable sólo indirectamente, llamando a estas operaciones. Hoy se usa el término objeto para tales módulos o variables encapsuladas a sí mismas. Lenguajes imperativos como Pascal y C han sido modificados (o añadidos) para que soporten el paradigma orientado a objetos para dar Delphi en el caso de Pascal y C++ en el caso de C.

Una de las bondades importantes de los lenguajes orientados a objetos es que las definiciones de los objetos pueden usarse una y otra vez para construir múltiples objetos con las mismas propiedades o modificarse para construir nuevos objetos con propiedades similares pero no exactamente iguales. El lenguaje orientado a objetos por excelencia es Smaltalk[2] desarrollado en Palo Alto Research Center durante los 1970's.

[1] Lenguaje de programación lógico e interpretado

[2] Primer sistema puro de objetos. Todo en Smalltalk es un objeto y toda la computación es desarrollada mediante mensajes que son enviados entre los objetos.

¿Pero qué es exactamente un lenguaje orientado a objetos? Los siguientes conceptos señalan las características generalmente aceptadas acerca de los lenguajes orientados a objetos.

- Objetos y clases son obviamente los conceptos fundamentales. Una clase es un conjunto de objetos que comparten las mismas operaciones.
- Objetos (o al menos referencia a objetos) deben ser valores de la clase base. Así, cualquier operación puede tomar un objeto como un argumento y puede devolver un objeto como resultado. De esta manera el concepto de clase de objetos está relacionado con el concepto de tipo de dato.
- Herencia es también vista como un concepto clave dentro del mundo de los objetos. En este contexto, la herencia es la habilidad para organizar las clases de objetos en una jerarquía de subclases y superclases y las operaciones dadas para una clase se pueden aplicar a los objetos de la subclase.

Luego de estudiar los paradigmas de programación, se decidió utilizar el paradigma Orientado a Objetos, por las ventajas que este le brinda al programador, y que van a permitir obtener un producto más robusto.

Una vez analizado los distintos paradigmas de programación y ver el más apropiado para dar solución al problema planteado, se debe hacer un estudio del lenguaje de programación a utilizar, el cual depende en gran medida de la selección anterior.

1.3. Lenguajes de Programación

Un lenguaje de programación, no es más que un lenguaje diseñado para describir el conjunto de acciones consecutivas que un equipo debe ejecutar. Permiten crear programas mediante un conjunto de instrucciones y operadores. Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Por lo tanto, un lenguaje de programación es un modo práctico para que los programadores puedan dar instrucciones a un equipo. (4)

1.3.1. Java

Es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90 con el que se pueden realizar cualquier tipo de programa. (5)

Ventajas

- Capacidad de que el código funcione sobre cualquier plataforma de software.
- Orientado a objetos y con encadenamiento múltiple.
- Es dinámico, los pequeños fragmentos de código Java se ensamblan en tiempo de ejecución en el programa y no en el momento en el que se escribe el código.
- Java es un lenguaje dinámico. La maquina virtual de java enlaza los programas java en tiempo de ejecución, eliminando la necesidad de enlazarlos con las librerías en tiempo de compilación.

Desventajas

- El rendimiento de una aplicación Java depende más de la eficiencia del compilador, o la JVM, que de las propiedades intrínsecas del lenguaje.
- El método que java utiliza para la creación del código, es ineficiente debido a que usa su máquina virtual para interpretar el código, lo que disminuye el rendimiento.

1.3.2. PHP 5.2.5

Fue creado en el año 1994 por Rasmus Lerdorf. Se conoce originalmente como Personal Home Pages y su primera versión salió en los comienzos de 1995. Es un lenguaje de "código abierto" interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor. Dentro de las operaciones que pueden realizarse con este lenguaje se encuentran: procesar la información de formularios, generar páginas con contenidos dinámicos o enviar y recibir cookies, entre otras. (6)

Existen tres campos en los que se usan scripts escritos en PHP.

- Scripts del lado del servidor. Este es el campo más tradicional y el principal foco de trabajo. Se necesitan tres cosas para que esto funcione. El intérprete PHP (CGI ó módulo), un servidor web y un navegador.

- Scripts en la línea de comandos. Puede crear un script PHP y correrlo sin ningún servidor web o navegador.

- Escribir aplicaciones de interfaz gráfica. Probablemente PHP no sea el lenguaje más apropiado para escribir aplicaciones gráficas, pero si conoce bien PHP, y quisiera utilizar algunas características avanzadas en programas clientes, puede utilizar PHP-GTK para escribir dichos programas.

Ventajas

- Puede ser utilizado en cualquiera de los principales sistemas operativos del mercado, incluyendo Linux, muchas variantes Unix (incluyendo HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X, RISC OS, entre otros.

- Soporta la mayoría de servidores web de hoy en día, incluyendo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape e iPlanet, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd y muchos otros.

- Posee una amplia documentación en su página oficial entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.

- Aunque no todas las características estándar de la programación orientada a objetos están implementadas en la versión actual de PHP, muchas bibliotecas y aplicaciones grandes (incluyendo la biblioteca PEAR) están escritas íntegramente usando programación orientada a objetos.

- No se encuentra limitado a resultados en HTML. Entre las habilidades de PHP se incluyen: creación de imágenes, archivos PDF y películas Flash. También puede presentar otros resultados, como XHTM y archivos XML. PHP puede autogenerar estos archivos y almacenarlos en el sistema de archivos en vez de presentarlos en la pantalla.

- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.

- La característica más potente y destacable de PHP es su soporte para una gran cantidad de bases de datos, entre ellas: mSQL, Direct MS-SQL, MySQL, ODBC, Oracle (OCI7 y OCI8), Ovrimos, PostgreSQL, y muchas más.

Desventajas

- No es escalable, debido a que no tiene medios propios de hacer programación distribuida.

La potencialidad de este lenguaje sobre los antes mencionados, al menos en el desarrollo de aplicaciones Web, es evidente, ya sea por ser "código abierto", multiplataforma o por el gran soporte de bases de datos con que cuenta. Por esto la selección de este para la implementación del sistema que defiende este trabajo de diploma.

La gran facilidad, ahorro de tiempo y código, que actualmente brindan los frameworks, es sin dudas un aspecto a considerar para llevar a cabo el desarrollo de una aplicación web. La utilización de estos está dada por el lenguaje de programación seleccionado, al ser PHP el lenguaje a utilizar para la implementación del sistema que defiende este Trabajo de Diploma, se muestra a continuación el estudio realizado sobre los mismos.

1.4. Frameworks para PHP

Se define un framework, como un conjunto de herramientas destinadas a la construcción de un determinado tipo de aplicaciones de manera generalista; es, en el ámbito informático, la estructura básica de una BD, proceso o programa. Su utilización simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, el empleo de un framework facilita la programación de aplicaciones, ya que encapsulan operaciones complejas en instrucciones sencillas.

1.4.1. CodeIgniter

CodeIgniter es un entorno de desarrollo abierto que permite crear webs dinámicas con PHP. Fue desarrollado por Rick Ellis para EllisLab, Inc. (7)

Su principal objetivo es ayudar a que los desarrolladores puedan realizar proyectos mucho más rápido que creando toda la estructura desde cero, proveyéndole un rico juego de librerías para tareas comúnmente necesarias, así como una interfaz simple y estructura lógica para acceder a esas librerías. CodeIgniter le permite creativamente enfocarse en su proyecto minimizando la cantidad de código necesaria para una tarea dada.

1.4.2. Symfony 1.2.8

Symfony es un completo framework diseñado para optimizar el desarrollo de las aplicaciones web mediante algunas de sus principales características. 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. (8)

Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony 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 Linux, como en plataformas Windows.

Ventajas

- Funciona correctamente en cualquier tipo de plataforma.
- Independiente del SGBD. 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 en la mayoría de casos, está más indicado para 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 BD, el controlador frontal y las acciones.
- Basado en la premisa de “convenir en vez de configurar”, en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- 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.

Desventajas

- Para su utilización se requiere de la generación de grandes cantidades de código y una configuración exhaustiva de antemano
- Precisa de una estructuración estricta de sus directorios que se establece por consola a la hora de crear módulos.

Después de analizar los frameworks de desarrollo mencionados anteriormente y teniendo en cuenta las características de estos y del software a desarrollar, así como el lenguaje que se va a usar para su desarrollo; se escogió el Symfony como framework de desarrollo para implementar la aplicación web que se propone en el presente Trabajo de Diploma.

1.5. Frameworks del lado del cliente

Un framework del lado del cliente ofrece funciones JavaScript para enviar peticiones al servidor. Algunos framework son muy robustos y proveen una biblioteca completa para construir aplicaciones web.

1.5.1. Dojo

Dojo es un framework que contiene APIs y widgets (controles) para facilitar el desarrollo de aplicaciones Web que utilicen tecnología AJAX. Contiene un sistema de empaquetado inteligente, los efectos de UI, drag and drop APIs, widget APIs, abstracción de eventos, almacenamiento de APIs en el cliente, e interacción de APIs con AJAX. Es conocido como "la navaja suiza del ejército de las bibliotecas JavaScript". (9)

Ventajas

- Resuelve asuntos de usabilidad comunes como pueden ser la navegación y detección del navegador, soportar cambios de URL en la barra de URLs para luego regresar a ellas (bookmarking).
- Tiene la habilidad de degradar cuando AJAX/JavaScript no es completamente soportado en el cliente.
- Proporciona una gama más amplia de opciones en una sola biblioteca JavaScript y es compatible con navegadores antiguos.

Desventajas

- La documentación que posee es muy escasa.

1.5.2. ExtJS 2.2

ExtJS es una biblioteca de JavaScript para el desarrollo de aplicaciones web interactivas usando tecnologías como AJAX, DHTML y DOM. (10)

Originalmente construida como una extensión de la biblioteca YUI, en la actualidad puede usarse como extensión para las bibliotecas jQuery y Prototype.

Sus características principales son:

- Gran desempeño.

- Componentes de interfaz de usuario personalizables.
- Variada documentación.
- Es una librería ligera y de alto rendimiento compatible con la mayoría de los navegadores.

Ventajas

- Permite crear aplicaciones complejas utilizando componentes predefinidos así como un manejador de layouts similar al que provee Java Swing.
- Existe un balance entre Cliente – Servidor. La carga de procesamiento se distribuye, permitiendo que el servidor, al tener menor carga, pueda manejar más clientes al mismo tiempo.
- Comunicación asíncrona. En este tipo de aplicación el motor de render puede comunicarse con el servidor sin necesidad de estar sujeta a un clic o una acción del usuario, dándole la libertad de cargar información sin que el cliente se dé cuenta.
- Eficiencia de la red. El tráfico de red puede disminuir al permitir que la aplicación elija que información desea transmitir al servidor y viceversa, sin embargo la aplicación que haga uso de la pre-carga de datos puede que revierta este beneficio por el incremento del tráfico.

Desventajas

- Necesita una plataforma. En este caso se depende de ExtJS para mostrar los componentes y hacer el render de la aplicación.
- El JavaScript no es tan rápido como se quisiera, sin embargo con la entrada de Google Chorme y el nuevo motor de Mozilla las cosas van mejorando.
- Problemas con los motores de búsqueda. Los motores de búsqueda indexan el contenido web estático por lo que los textos cargados de manera dinámica no serán encontrados.
- Accesibilidad. Estas aplicaciones tienen problemas con los programas de accesibilidad pues, al igual que los motores de búsqueda, no trabajan bien con texto cargado dinámicamente.

Una vez estudiados ambos frameworks se decidió ExtJs 2.2, teniendo en cuenta que sus componentes visuales, permitirán desarrollar una aplicación agradable y en muy poco tiempo.

Luego de analizar y de determinar el framework de desarrollo, el lenguaje de programación, el framework del lado del cliente, se hace necesario la selección de un Entorno de Desarrollo para la implementación del subsistema que se propone en este Trabajo de Diploma.

1.6. Entornos de Desarrollo

Un entorno de desarrollo integrado o IDE (acrónimo en inglés de Integrated Development Environment), es un programa informático compuesto por un conjunto de herramientas de programación.

Un IDE 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). (11)

1.6.1. Zend Studio for Eclipse

Es un completo Entorno de Desarrollo Integrado para el lenguaje de programación PHP. Está escrito en Java, y está disponible para las plataformas Microsoft Windows, Mac OS X y GNU/Linux. Además, sirve de editor de texto para páginas PHP, proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código. (12)

Ventajas

- No requiere la instalación previa de PHP ni del entorno de ejecución de Java.
- Soporte para PHP 4 y PHP 5.
- Resaltado de sintaxis, autocompletado de código, ayuda de código y lista de parámetros de funciones y métodos de clase.
- PhpDoc integrado.

- Plegado de código (comentarios, bloques de phpDoc, cuerpo de funciones y métodos e implementación de clases).
- Inserción automática de paréntesis y corchetes de cierre.
- Sangrado automático y otras ayudas de formato de código.
- Emparejamiento de paréntesis y corchetes.
- Detección de errores de sintaxis en tiempo real.
- Funciones de depuración: Botón de ejecución y traza, marcadores, puntos de parada (breakpoints), seguimiento de variables y mensajes de error del intérprete de PHP. Permite también la depuración en servidores remotos (requiere Zend Plataform).
- Soporte para gestión de grandes proyectos de desarrollo.
- Manual de PHP integrado.
- Soporte para control de versiones usando CVS o Subversion (a elección del desarrollador).
- Cliente FTP integrado.
- Soporte para navegación en bases de datos y ejecución de consultas del Lenguaje de Consultas Estructurado SQL (Structured Query Language).

Desventajas

- Es un IDE propietario.

1.6.2. Eclipse

Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. (13)

Ventajas

- Emplea plugins para proporcionar todas sus funcionalidades como plataforma.
- Dispone de un Editor de texto con resaltado de sintaxis.
- La compilación del código es en tiempo real.
- Tiene pruebas unitarias con JUnit.
- Control de versiones con CVS.
- Asimismo, a través de "plugins" libremente disponibles es posible añadir control de versiones con Subversion e integración con Hibernate.

1.6.3. PHP Development Tools (PDT)

PDT es el entorno de desarrollo basado en Eclipse específico para el desarrollo de aplicaciones con PHP. (14)

Entre sus ventajas:

- Resaltado, asistente y autocompletado de código.
- Integración con Xdebug. Soporte para el debug incremental del código de PHP.
- Posee explorador de archivos.
- Garantiza la depuración remota de aplicaciones.
- Extensos frameworks y APIs que permiten a los desarrolladores extender PDT.
- Aprendizaje intuitivo y fácil de la herramienta.
- Integración de phpdocumentator.

1.6.4. NetBeans 6.8 M2

NetBeans es una herramienta de código abierto para desarrolladores, que contiene las herramientas para que estos puedan crear aplicaciones desktop, enterprise, web, y aplicaciones móviles, con el lenguaje Java, así como también C/C++, PHP, JavaScript, Groovy, and Ruby. (15)

Ventajas

- Brinda completamiento automático de código PHP, así como coloreado de código sintáctico y semántico.
- Permite depurar el código usando Xdebug.
- Permite crear e importar Proyectos Symfony.
- Soporte para crear Aplicaciones añadiéndoles flags o parámetros.
- Soporte para PHPDoc.
- Consola de comandos de Symfony, muy útil para los usuarios de Windows, donde es un poco más complejo utilizar la consola.
- Soporta todos los tipos de aplicación Java y es el primero en soportar Java EE 6 Platform.
- Es un entorno de desarrollo multiplataforma que está disponible para las plataformas Windows, Linux, Mac OS X y Solaris.
- Tiene soporte para FTP y Subversion
- Producto libre y gratuito sin restricciones de uso.

Dadas las características, ventajas y soporte que brinda este potente Entorno de Desarrollo y sobre todo por ser su licencia “open source”, se eligió como IDE a utilizar para el desarrollo del sistema, además de que los antes mencionados (Zend Studio), tienen similares características pero son software privativos, estando en contra de la política seguida por nuestra Universidad para el desarrollo de software.

Para la continua manipulación de datos que trae consigo una aplicación, es preciso recurrir a un gestor de BD que se encargue de contener la información necesaria para el posterior uso de esta.

1.7. Sistemas Gestores de Bases de Datos

Los sistemas de gestión de bases de datos o SGBD (en inglés Database Management System, abreviado DBMS) son un tipo de software muy específico, dedicado a servir de interfaz entre la BD, el usuario y las aplicaciones que la utilizan.

Sistema Gestor de Base de Datos (SGBD)[3] que no es más que un sistema software que facilita la creación, el mantenimiento y uso de una BD electrónica, según el diccionario Wordnet de la Universidad de Princeton, en otros conceptos similares se tiene que es una colección de programas que se utiliza para almacenar, modificar y extraer información de una BD. De estos conceptos se deduce que un SGBD facilita las funciones de:

- Almacenar físicamente.
- Garantizar consistencia.
- Garantizar integridad.
- Atomicidad transaccional.
- Manejo de vistas a la información.

1.7.1 MySQL

MySQL es un SGBD relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB —desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009— desarrolla MySQL como software libre en un esquema de licenciamiento dual. (16)

[3] www.studiodog.com/glossary-d.html

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y el copyright del código está en poder del autor individual, MySQL es propietario y está patrocinado por una empresa privada, que posee el copyright de la mayor parte del código.

1.7.2. PostgreSQL- 8.3.4-1

PostgreSQL es un SGBD relacional orientada a objetos y libre, publicado bajo la licencia BSD.

Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una sola empresa sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo. Dicha comunidad es denominada el PGDG (PostgreSQL Global Development Group).

Algunas de sus principales características son, entre otras:

- ✓ 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.

Adicionalmente los usuarios pueden crear sus propios tipos de datos, los que pueden ser por completo indexables gracias a la infraestructura GiST de PostgreSQL. Algunos ejemplos son los tipos de datos GIS creados por el proyecto PostGIS.

Después del estudio realizado al respecto de las diferentes variantes actuales en el desarrollo de aplicaciones web, dígase lenguajes de programación, frameworks, SGBD, entornos de desarrollo, se concluye que para llevar a cabo la implementación del sistema, se empleará el lenguaje de programación PHP 5 en su versión 5.2, con el framework Symfony versión 1.2.8, sobre el entorno de desarrollo NetBeans 6.8 Milestone 2 para PHP, como framework del lado del cliente la librería ExtJS en su versión 2.2 y utilizando como SGBD PostgreSQL en su versión 8.3.4.1, por las facilidades que proveen en conjunto vistas con anterioridad en el cuerpo del capítulo. Todas las herramientas utilizadas tienen en común que son estandartes del software libre, líderes en su mayoría en este ámbito.

CAPITULO 2 Descripción y análisis de la solución propuesta

En este capítulo se abordará acerca de la implementación del sistema y la solución que se le propone en este trabajo de diploma, para lo cual, se realizará un valoración crítica del diseño propuesto por los analistas, teniendo en cuenta los requerimientos de la aplicación. Se hará referencia también a los componentes a reutilizar (clases o librerías previamente implementadas) y a las estrategias para su integración. Se describirán también los algoritmos no triviales a implementar y el análisis de la complejidad de los mismos, para finalmente describir las nuevas clases y operaciones necesarias para darle solución a la situación problemática.

2.1. Análisis de los diagramas.

En este epígrafe se procederá a analizar los diagramas que intervienen en la solución propuesta. Los cuales tienen una gran importancia para la realización de la implementación del sistema.

2.1.1. Análisis del diseño propuesto por los analistas.

A continuación se muestran los diagramas de clases de diseño propuestos por los analistas. Los mismos están acorde con los requerimientos del sistema a implementar y con el MVC propuesto por el framework Symfony.

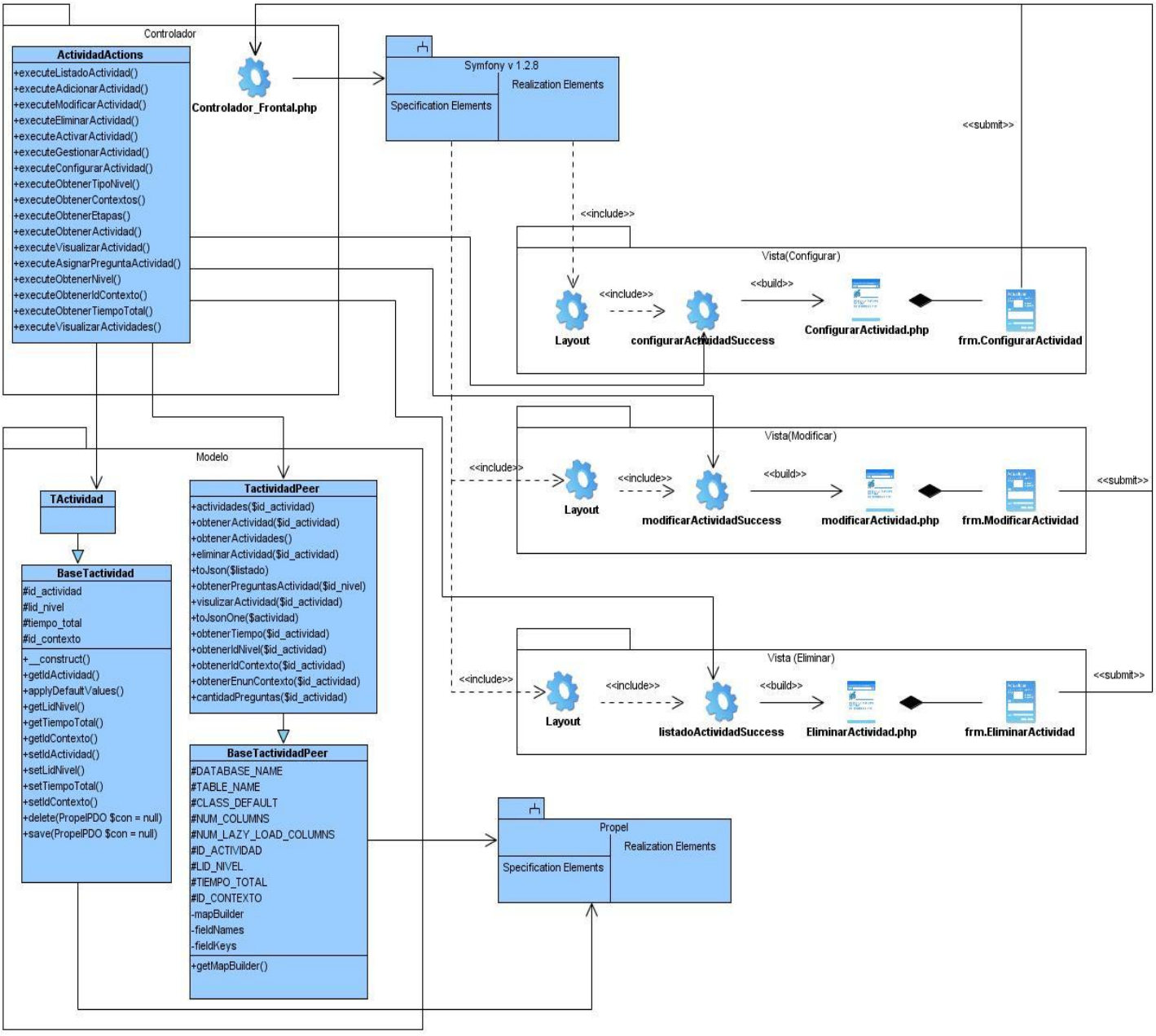


Ilustración 1: DCD Gestionar Actividad

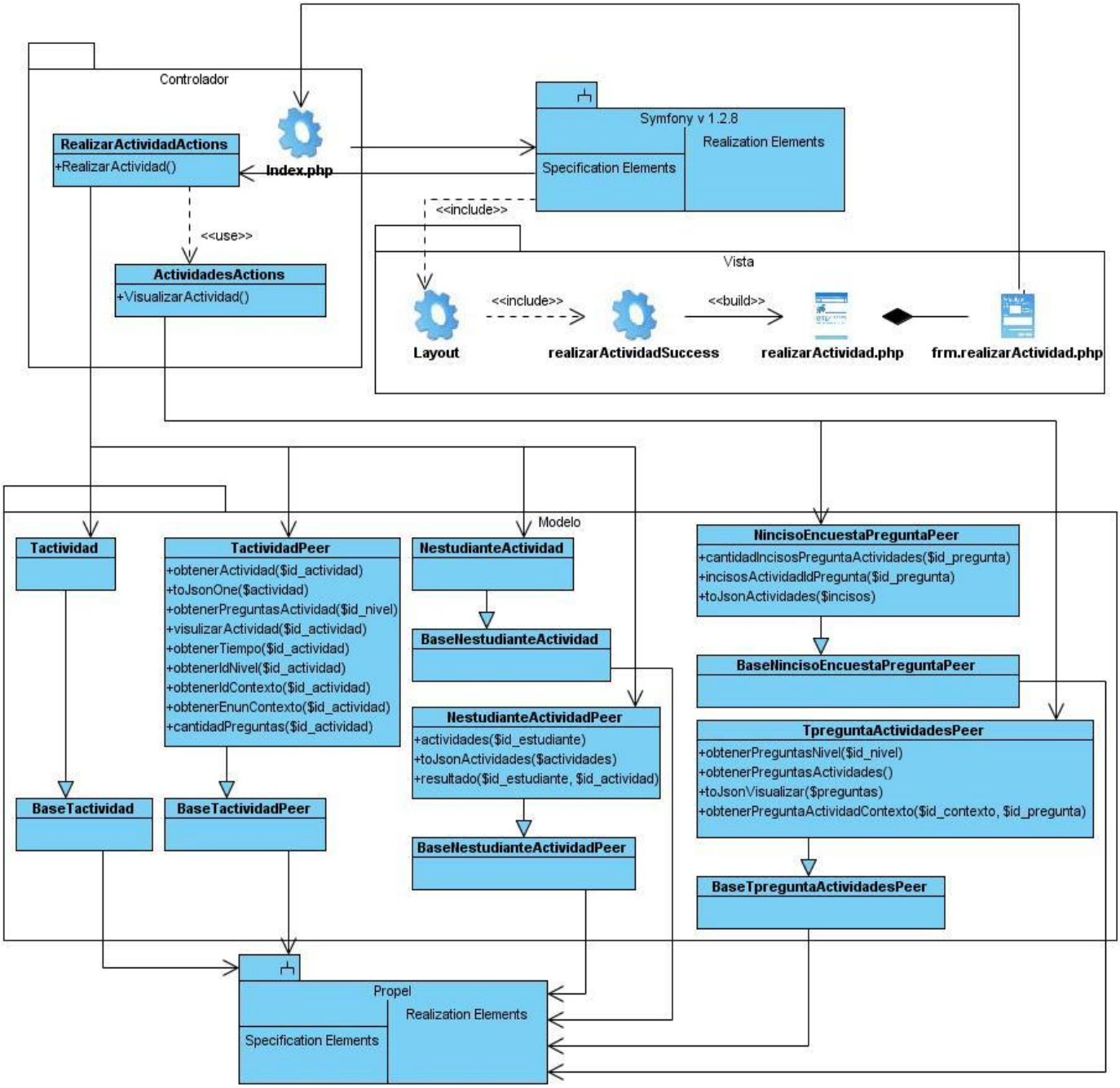


Ilustración 2: DCD Realizar Actividad

Posterior al análisis de la propuesta de diseño del analista, que tiene una gran repercusión puesto que decide qué se asume y qué se modifica para la implementación del sistema, se analizará de igual manera los diagramas de componentes.la reutilización de componentes o módulos ya existentes, que pueden ser reutilizados, así como las estrategias de integración.

2.1.2. Diagramas de componentes

A continuación se muestran los diagramas de componentes propuestos. Los mismos están acorde con los requerimientos del sistema a implementar y con el MVC propuesto por el framework Symfony.

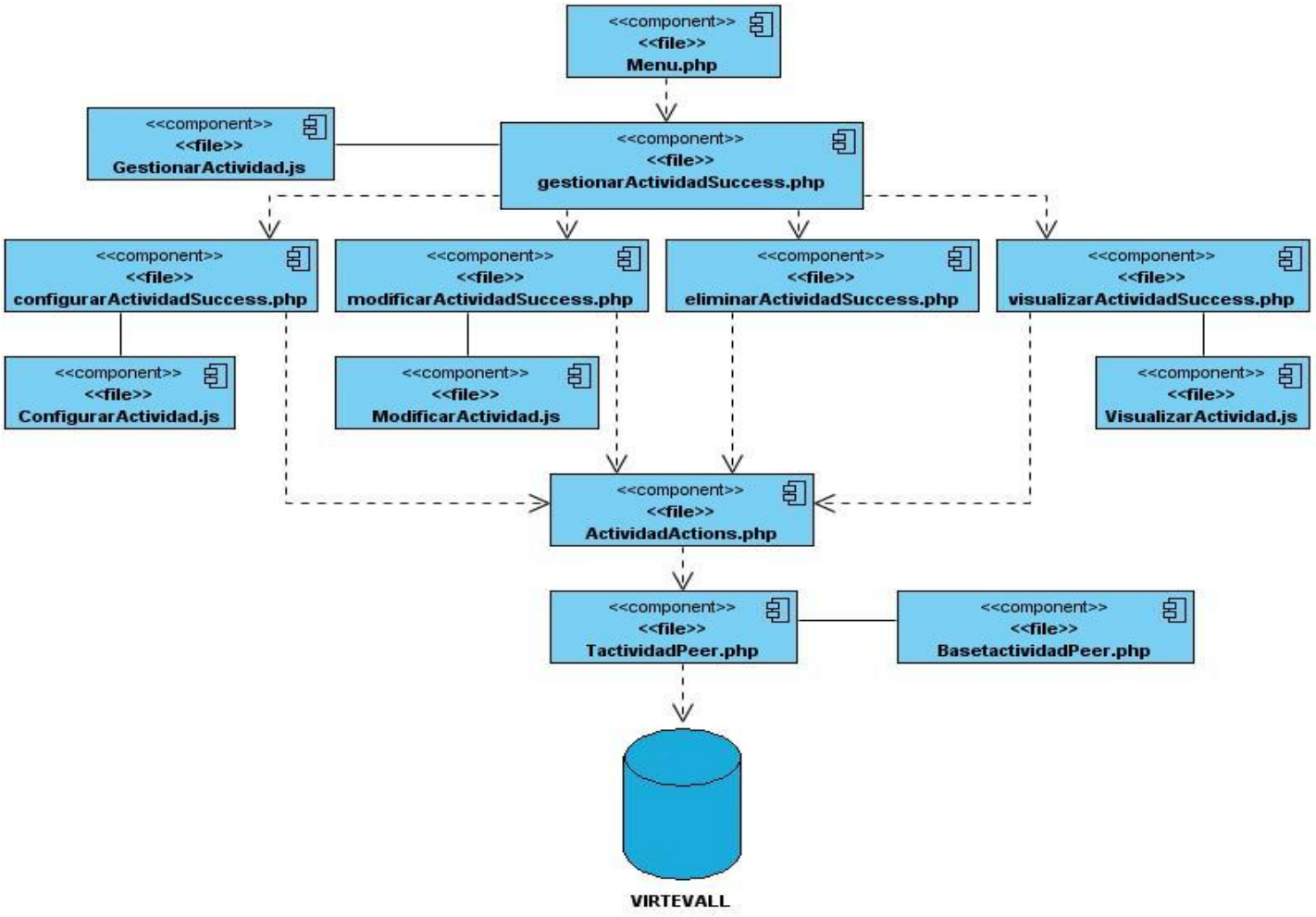


Ilustración 3: DC Gestionar Actividad

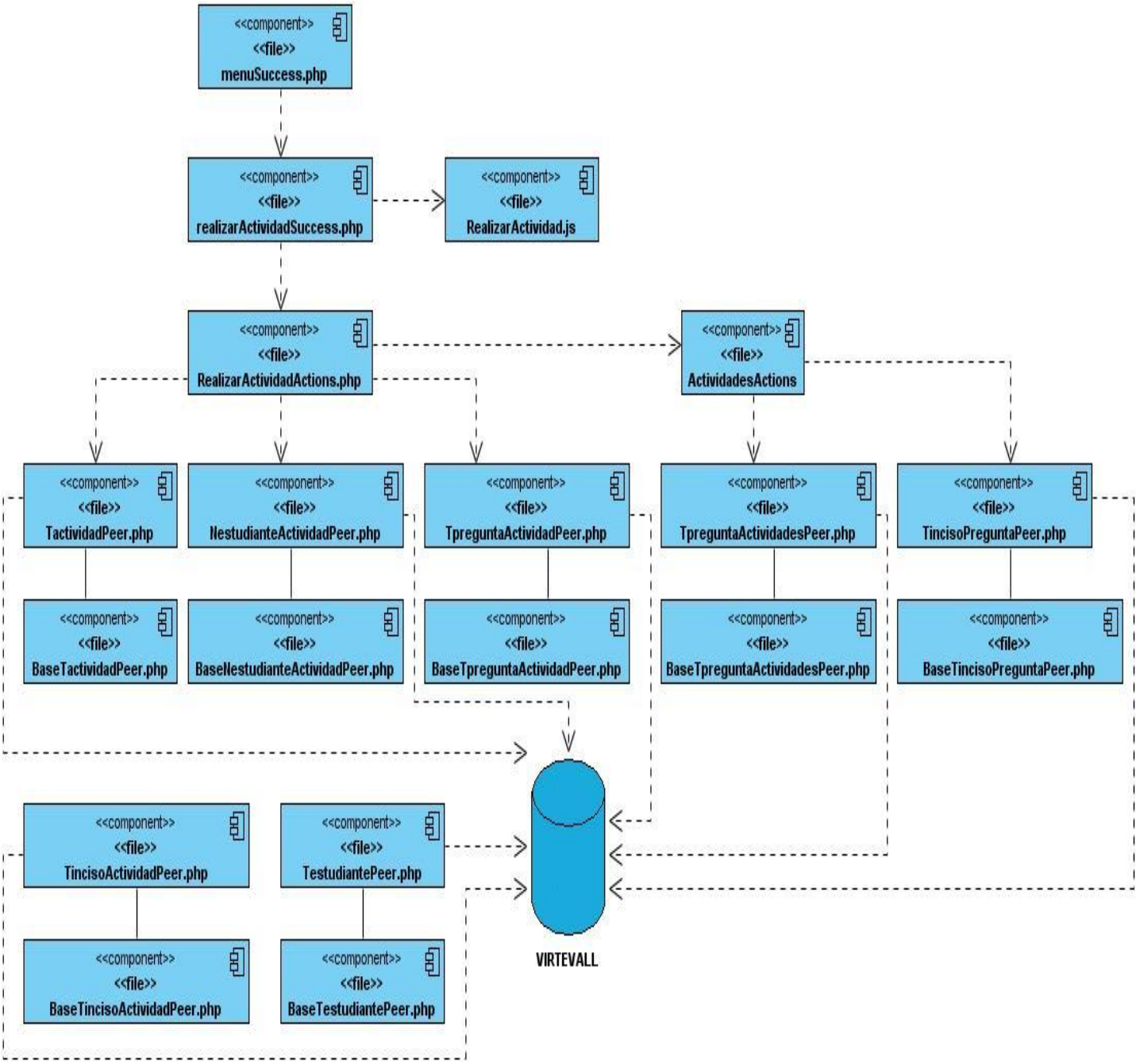


Ilustración 4: DC Realizar Actividad

Posterior al análisis de la propuesta de diseño del analista, y de los diagramas de componentes se analizará de igual manera la reutilización de componentes o módulos ya existentes, que pueden ser reutilizados, así como las estrategias de integración.

2.2. Análisis de posibles implementaciones, componentes o módulos ya existentes y que puedan ser rehusados. Estrategias de integración.

Symfony

Anteriormente se expuso que el framework de PHP seleccionado para el desarrollo de la aplicación fue Symfony, debido a que el mismo posee las características y funcionalidades necesarias que permitirían desarrollar de manera eficiente. Una de las características expuestas fue el uso del patrón de arquitectura Modelo-Vista-Controlador (MVC) que separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones.

MVC es una aproximación al software que separa la lógica de la aplicación de la presentación. En la práctica, permite que las páginas web contengan mínima codificación ya que la presentación es separada del código PHP.

El **modelo** representa la información con la que trabaja la aplicación, es decir, su lógica de negocio. Típicamente las clases de modelo contendrán funciones que para recuperar, insertar y actualizar información en la BD.

La **vista** transforma el modelo en una página web que permite al usuario interactuar con ella.

El **controlador** se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

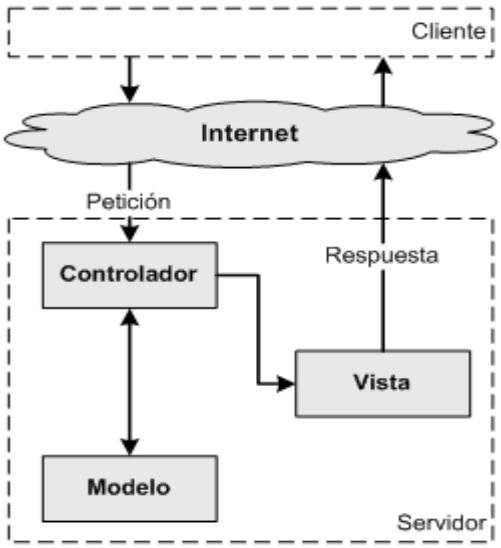


Ilustración 5: El patrón MVC

La implementación del MVC que realiza Symfony

El controlador frontal y el layout son comunes para todas las acciones de la aplicación. Se pueden tener varios controladores y varios layouts, pero solamente es obligatorio tener uno. El controlador frontal es un componente que sólo tiene código relativo al MVC, por lo que no es necesario crear uno, ya que Symfony lo genera de forma automática. (17)

Las clases de la capa del modelo también se generan automáticamente, en función de la estructura de datos de la aplicación. La librería Propel se encarga de esta generación automática, ya que crea el *esqueleto* o estructura básica de las clases y genera automáticamente el código necesario. La abstracción de la BD es completamente invisible al programador, ya que la realiza otro componente específico llamado Creole. Así, si se cambia el SGBD en cualquier momento, no se debe reescribir ni una línea de código, ya que tan sólo es necesario modificar un parámetro en un archivo de configuración.

La lógica de la vista se puede transformar en un archivo de configuración sencillo, sin necesidad de programarla.

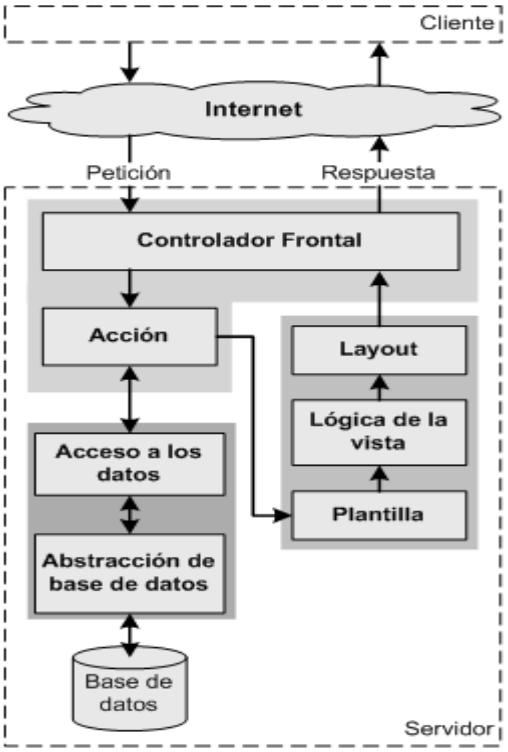


Ilustración 6: El flujo de trabajo de Symfony

PHPMailer

PHPMailer es una clase escrita en PHP que facilita el envío de correo, añadiendo facilidad en el envío de correos con adjuntos, en formato HTML y con diferentes codificaciones, soporte para imágenes embebidas, headers personalizados y además funciona con múltiples servidores de correo.

Típicamente para el envío de correo con PHP se utiliza la función ***mail ()***, pero esta función tiene varias limitaciones, por ejemplo que no soporta el envío de adjuntos. Entonces PHPMailer viene a facilitar este trabajo que de otra forma sería muy engorroso. Viene con un conjunto de métodos que ayudarán en el envió de emails.

Ldap

Librería que permite la conexión al LDAP desde PHP y gestionar usuarios, etc.

Métodos de la clase LDAP

```
public function Logearse ($domain, $usuario, $contrasenna)
```

Este método permite la conexión al servidor LDAP de la UCI, y verificar si el usuario forma parte del dominio, además retorna los datos del usuario en caso de que la conexión resulte satisfactoria.

Json

Gestiona la conversión de cadenas Json para uso en el JavaScript. JSON es un formato de intercambio de datos creado a partir de un subconjunto de la notación de literales de objetos en JavaScript. Aunque JavaScript acepta una sintaxis para valores literales muy flexible, es importante tener en cuenta que JSON posee reglas mucho más estrictas.

Se tiene ya una versión casi definitiva del diseño. Se ha decidido qué componentes utilizar y qué estrategia de integración tomar en consideración, durante la implementación del sistema. Por tanto en el epígrafe siguiente se analizarán los algoritmos que dan solución a algunos problemas, los cuales cobran cierto nivel de complejidad.

2.3. Descripción de los algoritmos no triviales a implementar. Análisis de complejidad de los mismos.

Los algoritmos que se analizan a continuación tienen un nivel de complejidad superior a la media, por lo que se hace necesario su estudio debido a que brindan la solución a problemáticas claves en la implementación del sistema.

Visualizar actividad

Consiste en visualizar una actividad con sus preguntas e incisos, teniendo en cuenta que una actividad no tiene incluida las preguntas ni los incisos; los cuales serán vinculados a las mismas en tiempo de ejecución teniendo en cuenta parámetros como el nivel, el contexto y la etapa de las actividades, preguntas e incisos.

Realizar actividad

Consiste en que un usuario realice una actividad, la solución que dé el usuario será registrada en la BD, además de esto se le irá mostrando al usuario los errores cometidos durante la solución de la misma.

Búsqueda

Consiste en la realización de una búsqueda de todos los elementos que en su nombre o descripción contengan la palabra especificada, siempre y cuando tenga al menos la cantidad de letras establecidas en el archivo de configuración, y en caso de que se especifique también un tipo de elemento, ya sea Incisos, Preguntas, Actividades, entonces la búsqueda se realizará solo en estos tipos de elementos.

Este cuenta además con un paginado dinámico permitiendo que los datos obtenidos se muestren según la cantidad especificada en el archivo de configuración, que contiene el número de elementos por página que deben mostrarse.

2.4. Descripción de las nuevas clases u operaciones necesarias.

Siguiendo el enfoque MVC del Symfony, las nuevas clases a describir serán de tipo Controladora, Modelo o Vista (Interfaz). Seguidamente se exponen las controladoras:

Nombre: ActividadesActions	
Tipo de clase: Controladora	
Atributos:	**Tipo:**
-	-

Funciones:	
Nombre:	adicionarActividad
Descripción:	Permite adicionar una actividad
Nombre:	listadoActividades
Descripción:	Muestra un listado con las actividades existentes
Nombre:	modificarActividad
Descripción:	Modifica los elementos de una actividad
Nombre:	eliminarActividad
Descripción:	Permite eliminar una actividad
Nombre:	activarActividad
Descripción:	Permite activar una actividad
Nombre:	obtenerActividad
Descripción:	Permite obtener una actividad dado el identificador de la misma
Nombre:	obtenerTiempoTotal
Descripción:	Permite obtener el tiempo que dura una actividad
Nombre:	visualizarActividad
Descripción:	Permite visualizar una actividad determinada

Tabla 1: Clase Control ActividadesActions

Nombre: PreguntaActions	
Tipo de clase: Controladora	
Atributos:	**Tipo:**
-	-
Funciones:	
Nombre:	adicionarPreguntaActividad

Descripción:	Permite adicionar una pregunta
Nombre:	listadoPreguntasActividades
Descripción:	Muestra un listado con todas las preguntas que puede tener una actividad
Nombre:	modificarPreguntaActividad
Descripción:	Modifica los elementos de una pregunta
Nombre:	eliminarPreguntaActividad
Descripción:	Permite eliminar un inciso de una actividad
Nombre:	activarPregunta
Descripción:	Permite activar una pregunta
Nombre:	obtenerRespuesta
Descripción:	Permite obtener la respuesta de un inciso de una actividad
Nombre:	obtenerNivel
Descripción:	Permite obtener el nivel de un inciso de una actividad

Tabla 2: Clase Control PreguntaActions

Nombre: IncisoActions	
Tipo de clase: Controladora	
Atributos:	**Tipo:**
-	-
Funciones:	
Nombre:	adicionarIncisoActividades
Descripción:	Permite adicionar un inciso a una actividad
Nombre:	listadoIncisosActividades
Descripción:	Muestra un listado con todos los incisos de una actividad

Nombre:	modificarIncisoActividad
Descripción:	Modifica los elementos de un inciso de una actividad
Nombre:	eliminarIncisoActividad
Descripción:	Permite eliminar un inciso de una actividad
Nombre:	activarInciso
Descripción:	Permite activar un inciso de una actividad
Nombre:	obtenerRespuesta
Descripción:	Permite obtener la respuesta de un inciso de una actividad
Nombre:	obtenerNivel
Descripción:	Permite obtener el nivel de un inciso de una actividad

Tabla 3: Clase Control IncisoActions

Nombre: LoginActions	
Tipo de clase: Controladora	
Atributos:	**Tipo:**
-	-
Funciones:	
Nombre:	login
Descripción:	Permite autenticar un usuario
Nombre:	adicionarUsuario(Tusuario $usuario)
Descripción:	Permite adicionar un usuario
Nombre:	logout
Descripción:	Permite terminar la sesión de un usuario
Nombre:	cambiarClave
Descripción:	Permite cambiar la clave de un usuario en la BD.

Tabla 4: Clase Control LoginActions

Nombre: ActividadPeer	
Tipo de clase: Modelo	
Atributos:	**Tipo:**
-	-
Funciones:	
Nombre:	obtenerActividad($id_actividad)
Descripción:	Permite obtener los datos de una actividad
Nombre:	eliminarActividad($id_actividad)
Descripción:	Permite eliminar una actividad
Nombre:	visulizarActividad($id_actividad)
Descripción:	Permite visualizar una actividad
Nombre:	obtenerTiempo($id_actividad)
Descripción:	Permite obtener el tiempo que dura una actividad
Nombre:	obtenerIdNivel($id_actividad)
Descripción:	Permite obtener el nivel de una actividad
Nombre:	obtenerIdContexto($id_actividad)
Descripción:	Permite obtener el contexto de una actividad
Nombre:	toJson($listado)
Descripción	Convierte y devuelve en formato Json un listado de actividades
Nombre:	toJsonOne($actividad)
Descripción	Convierte y devuelve en formato Json una actividad

Tabla 5: Clase del Modelo ActividadPeer

Nombre: IncisoPreguntaPeer
Tipo de clase: Modelo

Atributos:	**Tipo:**
-	-
Funciones:	
Nombre:	obtenerInciso($id_inciso)
Descripción:	Permite obtener los datos de un inciso
Nombre:	eliminarInciso($id_inciso)
Descripción:	Permite eliminar un inciso
Nombre:	respuestaInciso($id_inciso)
Descripción:	Permite obtener la respuesta de un inciso
Nombre:	nivelInciso($id_inciso)
Descripción:	Permite obtener el nivel de un inciso
Nombre:	obtenerIdNivel($id_actividad)
Descripción:	Permite obtener el nivel de una actividad

Tabla 6: Clase del Modelo IncisoPreguntaPeer

Nombre: PreguntaActividadesPeer	
Tipo de clase: Modelo	
Atributos:	**Tipo:**
-	-
Funciones:	
Nombre:	obtenerPreguntasNivel($idnivel)
Descripción:	Permite obtener los datos de todas las preguntas con el nivel especificado
Nombre:	obtenerPreguntas()
Descripción:	Permite obtener todas las preguntas
Nombre:	obtenerPregunta($idpregunta)
Descripción:	Permite obtener los datos de una pregunta especificada.

Tabla 7: Clase del Modelo PreguntaActividadesPeer

Nombre: PreguntaPeer	
Tipo de clase: Modelo	
Atributos:	**Tipo:**
-	-
Funciones:	
Nombre:	eliminarPregunta($id_pregunta)
Descripción:	Permite eliminar una pregunta
Nombre:	obtenerEnunciado($id_pregunta)
Descripción:	Permite obtener el enunciado de una pregunta
Nombre:	toJsonPreguntaActividadNivel($preguntas)
Descripción:	Permite convertir y obtener en formato Json las preguntas con un nivel determinado
Nombre:	toJsonPreguntaActividad($preguntas)
Descripción:	Permite convertir y obtener en formato Json las preguntas.

Tabla 8: Clase del Modelo PreguntaPeer

2.5. Patrones de Diseño utilizados en la implementación

2.5.1. Patrones GRASP

Experto: Este es uno de los más utilizados, puesto que Propel es la librería externa que utiliza Symfony para realizar su capa de abstracción en el modelo, encapsula toda la lógica de los datos y son generadas las clases con todas las funcionalidades comunes de las entidades.

Alta Cohesión: En cada clase Actions se definen las acciones para las plantillas, además estas colaboran con otras para realizar diferentes operaciones, se instancian objetos, se acceden a las properties, es decir en una Actions se utilizan diferentes funcionalidades estrechamente relacionadas entre sí, lo que proporciona un software flexible ante los cambios.

Controlador: Todas las peticiones Web son manejadas por un solo controlador frontal (sfActions), que es el punto de entrada único de toda la aplicación en un entorno determinado.

Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para asociar el nombre de una acción y el nombre de un módulo con la URL entrada por el usuario.

2.5.2. Patrones GOF

- ✓ En la categoría Creacionales:

Singleton (Instancia única): Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. En las acciones se usan los métodos ->getRequest (), ->getUser (), esto se debe a que, en la acción, el método getContext (), guarda una referencia a todos los objetos del núcleo de Symfony, estos métodos pueden ser accedidos desde la vista y desde el controlador, solo varía la forma de llamarlos.

- ✓ En la categoría Estructurales:

Decorator (Envoltorio): Añade funcionalidad a una clase, dinámicamente. En cada archivo layout.php se define el código HTML común de cada vista, evitando que este sea repetido en cada página.

2.6. Estándares de Codificación

Es prudente establecer estándares de codificación para todos los programadores. Estos estándares consisten en estilos de codificación a la hora de escribir el código. Los aspectos para los que generalmente se establecen estándares son los siguientes:

- Identificadores.
- Indentación.
- Líneas y espacios en blanco.
- Comentarios.

En cada grupo de desarrollo se definen cuales serán los aspectos a estandarizar y que estilos se aplicarán a cada uno de ellos. El cumplimiento de estándares hace que todo el código lleve el sello personal del programador y en caso de ser varios los programadores pues se busca que todo el código parezca que ha sido implementado por la misma persona. De esta manera se consigue mayor legibilidad y facilidad de mantenimiento. Los estándares deben responder además a acciones prácticas

que acomoden al programador. A continuación se definirá que estándares usar para la actividad de implementación correspondiente a este trabajo. Cabe destacar que estos estándares se definen teniendo en cuenta el estilo personal del programador, las características propias del lenguaje de programación, los recursos que se utilizarán y el tipo de programa que se debe implementar.

2.6.1. Identificadores

En el caso de los identificadores existen estilos definidos mundialmente como el lowerCamelCase y el UpperCamelCase. Cada palabra interna en identificadores compuestos comienza con mayúsculas para ambos estilos, además ocurre que no se colocan caracteres de separación entre las palabras que conforman un identificador compuesto en ninguno de los dos casos. Para el primero, el identificador comienza con minúscula y para el segundo, el identificador comienza con mayúscula.

- Espacio de nombre: Para los espacios de nombres se escogió el UpperCamelCase.
- Clase: Para las clases se escogió el UpperCamelCase.

```
class ActividadesActions extends sfActions
```

- Función: Para las clases se escogió el lowerCamelCase.

```
public function getIdActividad()
```

- Variable

```
protected $id_actividad;
```

2.6.2. Indentación

La Indentación es una práctica de programación que consiste en comenzar a escribir cada línea de código a diferentes distancias desde el borde izquierdo del área de texto del editor. Esta distancia está determinada por la jerarquía que se forma al introducir sentencias dentro de bloques de estructuras. Esto brinda mayor legibilidad y entendimiento para el programador e igualmente depende del propio estilo de cada persona, del lenguaje y tipo de programa que se implementa. Se definió que la indentación se hará agregando dos *tab* al inicio de la línea que se desee escribir. Se escribirá solo una sentencia por línea de código y en el caso de cortar las líneas, se hará luego de una coma o antes de

un operador. La sección de la derecha de la línea que se corte se ubicará en la línea siguiente indentada al nivel de la expresión correspondiente en la línea superior.

```
1  <?php
2
3  /**...
11 class ActividadesActions extends sfActions
12 {
13  /**...
18   //Gestionar Actividad
19     public function executeListadoActividad()
20     {...}
23
24     public function executeAdicionarActividad()
25     {...}
48
49     public function executeModificarActividad()
50     {...}
73
74     public function executeEliminarActividad()
75     {...}
87
88     public function executeActivarActividad()
89     {...}
93
```

Ilustración 7: Ejemplo de Indentación

2.6.3. Llaves

Existe diversidad de criterios en cuanto a la ubicación de las llaves que delimitan el cuerpo de los bloques de código en los lenguajes que contienen este tipo de estructuras. Algunos programadores prefieren hacerlo ubicando la llave de apertura inmediatamente detrás de la línea cabecera del bloque mientras otros apuestan por ubicarlas de forma solitaria en la línea siguiente a la línea cabecera. Para este último estilo existen además diferencias en cuanto al nivel de indentación de las mismas. Algunos lo hacen al nivel de la línea cabecera y otros al nivel de las líneas del cuerpo del bloque. Para este trabajo: Las llaves de apertura de las funciones se colocarán solitarias en la línea siguiente e indentadas al nivel de la línea cabecera del bloque y las llaves de aperturas de las estructuras *for, if, while, else*, se colocarán inmediatamente detrás de la línea cabecera del bloque. Las llaves de cierre se colocarán solitarias en la línea que sigue a la última línea dentro del bloque e indentadas al nivel de la línea cabecera del bloque. Es prudente señalar que este estilo agrega más líneas de código al programa al ubicar las llaves solitarias en una línea pero a su vez se gana en legibilidad del código.

```
public function executeAdicionarPreguntaActividad()
{
if ($this->getRequest()->getMethod () != sfRequest::POST){
            return sfView::SUCCESS;
    }
    else {
```

Ilustración 8: Ejemplo de uso de llaves

2.6.4. Líneas y espacios en blanco

Para mejorar la legibilidad y organización del código muchas veces se utilizan líneas en blanco para separar segmentos de código que pueden corresponder a clases, funciones, declaraciones, implementaciones, comentarios, bloques o sencillamente secciones críticas que se deseen despejar. Así mismo sucede con los espacios en blanco cuando se utilizan para separar elementos dentro de las sentencias de código. En ocasiones se separan con espacios cada operador de su respectivo operando, paréntesis, identificadores, símbolos y algunos lenguajes exigen que se separen las palabras propias del vocabulario de las adyacentes para ser comprendidas por los compiladores. En este trabajo se ha definido emplear líneas en blanco:

- Entre funciones

```
public function executeObtenerNivel()
{
    $nivel = TactividadPeer::ObtenerIdNivel($this->getUser()->getAttribute('
    return $this->renderText($nivel);
    die();
}

public function executeObtenerIdContexto()
{
    $cont = TactividadPeer::ObtenerIdContexto($this->getUser()->getAttribute
    return $this->renderText($cont);
    die();
}

public function executeObtenerTiempoTotal()
{
    $time = TactividadPeer::ObtenerTiempo($this->getUser()->getAttribute('id
    return $this->renderText($time);
```

Ilustración 9: Ejemplo 1 de espacios en blanco

- Entre declaraciones de variables e implementaciones dentro del cuerpo de las funciones

Se colocarán espacios en blanco:

- Entre las palabras reservadas y los elementos adyacentes a las mismas.

```
protected $id_actividad;
```

- Después de las comas en la lista de argumentos de las funciones.
- Después de cada punto y coma (;) en las estructuras for.

```
for ($i = 0; $i < count($ids); $i++){
    $incisos[$i] = TincisoEncuestaPeer::ObtenerInciso($ids[$i]);
}
```

Ilustración 10: Ejemplo 2 de espacios en blanco

2.6.5. Comentarios

El uso de comentarios durante la codificación ha demostrado que es beneficiosa por varias razones:

- Ayuda al programador a entender cada elemento o sección de código.
- Hace más fácil el proceso de adaptación del código durante su reutilización.
- Sirve de guía en los casos en que varios programadores trabajen sobre las mismas secciones del código.
- Disminuye el esfuerzo de análisis ya que el lenguaje natural es más legible que cualquier lenguaje de programación.

Todo esto se aprecia claramente cuando se escribe gran cantidad de código en largos intervalos de tiempo donde generalmente el o los programadores olvidan lo que se pensó en un momento. Los comentarios se pueden utilizar para varios fines:

- Para explicar el propósito de las funciones.
- Para explicar las características fundamentales de las clases.
- Para sintetizar las acciones de los algoritmos complejos.
- Para aclarar los datos que representan las variables.

- Para dividir secciones de código en dependencia de los diferentes contextos y funciones.
- A veces se usan comentarios temporales para recordar cosas que faltan, cosas que se deben modificar o analizar en otro momento.

Es necesario tener en cuenta algunos detalles al escribir comentarios:

- La capacidad de síntesis.
- El uso de lenguaje técnico.
- No repetir exactamente paso por paso lo que hace el algoritmo sino expresar un resumen de su propósito.
- Usar un estilo uniforme de comentario definido en estándares para todo el equipo.

En este trabajo se decidió colocar los comentarios encima de la línea a la que se le quiera aplicar y encima de la línea cabecera de los bloques. La Indentación se hará al nivel de la línea en cuestión. Se utilizarán en funciones y clases. Se puede utilizar en algoritmos no triviales y secciones de diferentes contextos dentro de los métodos.

Después de realizar una valoración crítica del diseño propuesto por el analista, se ha podido arribar a un diseño acorde con las restricciones del sistema, que deben tenerse en consideración a la hora de implementarlo.
También se realizó un análisis a un conjunto de librerías y clases previamente implementadas a reutilizar, que repercuten en la dinámica de implementación porque brindan facilidades que es solamente cuestión de utilizar, entre ellas la estructura del framework de PHP a emplear, Symfony.

De igual manera se estudiaron los algoritmos no triviales presentes en la solución, realizándole a cada uno un análisis de su complejidad y se describieron las nuevas clases y funciones necesarias para solucionar el problema, estas clases según el patrón de diseño que implementa Symfony, se clasificaron en tres grupos básicos: controladoras, modelos y vistas (interfaces). Todas estas cuestiones tienen una repercusión directa en la consumación de la aplicación.

CAPITULO 3 Validación de la solución propuesta

La prueba del software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación. El objetivo de la etapa de pruebas es garantizar la calidad del producto desarrollado. Es una actividad en la cual un sistema o componente es ejecutado bajo condiciones o requerimientos específicos, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente.

En el presente capítulo se hace una búsqueda de las técnicas de prueba de software que permitan comprobar la validez de la solución propuesta. Se detallan los casos de prueba con sus objetivos y alcance. Se aborda el tema relacionado con las herramientas empleadas en el proceso de desarrollo y se analizan los resultados obtenidos.

3.1. Diseño de las pruebas unitarias y funcionales.

Las pruebas unitarias permiten probar cada unidad independiente del software. Actúan esencialmente sobre el código fuente y sobre los elementos básicos de la interfaz de cada módulo. Los casos de prueba que se generan durante las pruebas de unidad deben estar encaminados a verificar los siguientes elementos: (18)

- Interfaz: “Se prueba la interfaz del módulo para asegurar que la información fluye de forma adecuada hacia y desde la unidad de programa que está siendo probada”.
- Estructuras de datos locales: “Se examinan las estructuras de datos locales para asegurar que los datos que se mantienen temporalmente, conservan su integridad durante todos los pasos de ejecución del algoritmo”.
- Condiciones límites: “Se prueban las condiciones límites para asegurar que el módulo funciona correctamente en los límites establecidos como restricciones de procesamiento”.
- Caminos independientes: “Se ejercitan todos los caminos independientes de la estructura de control para asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez”.
- Caminos de manejo de errores: “Se prueban todos los caminos de manejo de errores”.

Con las pruebas unitarias es posible aislar una parte del código de manera que pueda ser analizado. Un ejemplo de esto es evaluar las funciones o métodos, a los cuales se les realiza una entrada de datos para obtener los datos de salida correctos. Este tipo de pruebas validan la forma en la que las funciones y métodos trabajan en cada caso particular. Las pruebas funcionales no solo validan la transformación de una entrada en una salida, sino que validan una característica completa. De modo que las pruebas funcionales validan procesos y requieren de un escenario. En Symfony se deberían crear pruebas funcionales para todas las acciones.

Estas pruebas simulan la navegación del usuario, realizan peticiones y comprueban los elementos de la respuesta tal y como lo haría manualmente un usuario para validar que una determinada acción hace lo que se supone que debe hacer. En las pruebas funcionales se ejecuta un escenario correspondiente a lo que se denomina un "caso de uso".

3.2. Tipos de pruebas de software

La calidad de un sistema está determinada, entre otras cosas, por la coincidencia entre lo que se programó y los requisitos establecidos en la primera fase. Para comprobar el grado de cumplimiento de estos requisitos se usan las pruebas del sistema. La prueba de unidad es la prueba enfocada a los elementos probables más pequeños del software, consiste en una prueba estructural (o caja blanca), lo cual requiere conocer el diseño interno de la unidad puesto que verifica la lógica interna y el flujo de datos; y una prueba de especificación (de caja negra), basada sólo en la especificación del comportamiento externamente visible de la unidad.

Estas definen un conjunto amplio de acciones de comprobación que abarcan todas las características que determinan la calidad de un software. Se comprueban las funcionalidades diseñando casos de prueba que definen cómo proceder. Estos casos de prueba incluyen los juegos de datos a usar que son los válidos o esperados y los no válidos o no esperados por el programa. Las pruebas se deben aplicar durante todo el ciclo de vida del software e invariablemente se le debe dedicar una gran parte del esfuerzo total del desarrollo. Se deben planificar correctamente desde el inicio y establecer qué hacer, cómo hacer, quién va a hacer y en qué condiciones hacer las comprobaciones.

El objetivo de los casos de prueba es forzar al máximo el sistema en los puntos críticos para encontrar fallos y detectar defectos.

Se debe escoger los tipos de prueba que se adapten mejor al sistema que se va a probar. Para esto se debe tener en cuenta el lenguaje de programación, el proceso de desarrollo, las características de los desarrolladores, el tipo de funcionalidad que se implementa, la plataforma en que se ejecutan los procesos, los errores más importantes, si la aplicación es de escritorio o web, si realiza conexiones a bases de datos entre otras observaciones.

3.2.1. Pruebas de Caja Blanca

Las pruebas de Caja Blanca se nombran de esta forma porque revisan la parte interna del software, específicamente sobre el código fuente. Se basan en el examen minucioso de los detalles procedimentales. Se comprueban los caminos lógicos del sistema generando casos de prueba que ejerciten las estructuras condicionales y los bucles. Existen varios métodos que analizan diferentes partes del programa y se complementan entre sí para garantizar la calidad del sistema. La técnica del camino básico se utiliza para comprobar la complejidad lógica de un diseño procedimental, permite diseñar casos de prueba para cubrir todas las sentencias de un programa a partir de la obtención de un conjunto de caminos independientes.

La complejidad ciclomática, como resultado fundamental de estas pruebas, acota la cantidad mínima de casos de prueba que se deben ejecutar. La prueba de las Condiciones es un método que se encamina hacia la ejercitación de las condiciones. Se basa en el principio de que si un conjunto de casos de prueba es capaz de ejercitar todas las condiciones contenidas en un bloque de código, este mismo conjunto serviría para encontrar más errores en el programa que no tengan que ver directamente con las condiciones. La prueba del Flujo de Datos verifica la validez en el uso de las variables para manipular los datos de la aplicación. Selecciona los casos de prueba atendiendo a las definiciones y los usos de las variables. El procedimiento indica que se debe encontrar las sentencias donde se define cada variable y las sentencias donde se hace uso de las mismas. Luego se encuentran las —cadenas de definición - uso que representan el ciclo de vida de las variables y se diseñan casos de prueba que las ejecuten en su totalidad. La prueba de los Bucles se centra en la validez de las estructuras cíclicas o bucles. El objetivo es probar el comportamiento de estas estructuras en sus valores límites de iteración. Los bucles se clasifican en cuatro tipos: Bucles simples, Bucles anidados, Bucles concatenados y Bucles no estructurados. Aunque la esencia es la misma, cada tipo se prueba de forma diferente. De lo anterior pudiera pensarse que las pruebas de Caja Blanca logran enmendar todos los errores del programa. La

desventaja de estas es entonces de tipo logística ya que resulta imposible abarcar todo el código fuente de un sistema medianamente grande. El tiempo necesario para realizarlas sería considerable y se torna compleja su aplicación sobre algoritmos críticos.

3.2.2. Pruebas de Caja Negra

Las Pruebas de Caja Negra deben su nombre a los elementos que estas revisan y las condiciones en que se hace la revisión. Estas se basan en los requerimientos funcionales del sistema y se llevan a cabo desde el exterior de la aplicación. Este tipo de prueba es importante a la hora de medir el grado de cumplimiento de los requerimientos solicitados por el cliente y se aplican sobre la interfaz de la aplicación observando las respuestas del sistema antes determinadas acciones y los datos de salida para determinados datos de entrada.

3.3. Herramientas para la realización de pruebas

Cuando se emplea php como lenguaje de programación existe diversidad de frameworks para crear pruebas unitarias y funcionales, siendo los más usados PhpUnit y SimpleTest. Symfony incluye su propio framework para pruebas unitarias llamado Lime. El mismo utiliza la librería Test::More de Perl y es compatible con TAP. Lo que significa que los resultados de las pruebas se muestran con el formato definido en el "Test Anything Protocol", creado para facilitar la lectura de los resultados de las pruebas. El Lime presenta además una serie de ventajas dentro de las cuales resaltan por su importancia las siguientes: (18)

- Ejecuta los archivos de prueba en un entorno independiente para evitar interferencias entre las diferentes pruebas.
- Las pruebas son fáciles de leer y sus resultados también lo son. En los sistemas operativos que lo soportan, los resultados utilizan diferentes colores para mostrar de forma clara la información más importante.
- Está escrito con PHP, es muy rápido y está bien diseñado internamente. Consta únicamente de un archivo llamado lime.php y no tiene ninguna dependencia.

A pesar de estas ventajas Lime no resulta un framework suficiente para el proceso de prueba, ya que en caso que se necesite realizar validaciones del lado del cliente, no brinda la posibilidad de simular el comportamiento de JavaScript, el cual es un lenguaje que es empleado en el proceso de validación.

Con el objetivo de obtener un producto confiable, que funcione correcta y eficientemente se decidió realizarle ambos tipos de prueba, para esto se diseñaron casos de pruebas.

3.4. Diseño de Casos de Prueba de Caja Blanca aplicados

El objetivo fundamental del diseño de casos de prueba es conseguir un conjunto de pruebas que tengan la mayor probabilidad de encontrar los defectos del software.

Un caso de prueba es un conjunto de entradas de pruebas, condiciones de ejecución y resultados esperados desarrollados para verificar una función esperada.

Descripción del Caso de Uso Adicionar Actividad

Permite Adicionar una actividad

```
public function Adicionar($actividad) {
 if ($this->obj->ok($actividad['nombre'] != '  ' , 'El parámetro nombre está correcto.')) {
  if ($this->obj->ok($actividad['id_nivel'] != '  ' , 'El parámetro descripción está correcto.')) {
   if($this->obj->ok($this->existeNivel($actividad['id_nivel']) != false, 'El nivel existe')) {
    if ($this->obj->ok($actividad['tiempo_total'] != ' ', 'El parámetro descripción está correcto.')) {
     if ($this->obj->ok($actividad['id_contexto'] != ' ', 'El parámetro descripción está correcto.')) {
      if($this->obj->ok($this->existeContexto($actividad['id_contexto']) != false, 'El contexto existe')) {
            $act = new  Tactividad();
            $act->setNombre($actividad['nombre']);
            $act->setLidNivel($actividad['id_nivel']);
            $act->setIdContexto($actividad['id_contexto']);
            $act->setTiempoTotal($actividad['tiempo_total']);
            $act->save();
            $this->obj->diag('La actividad fue adicionada');
           }}}}}}}
```

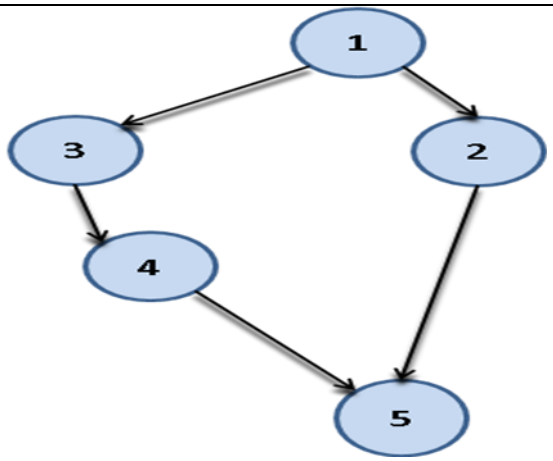
Complejidad ciclomática V (G) = 2	
Posibles Caminos 1-3-4-5 1-2	

Tabla 9: Escenario Adicionar Actividad

Definir Juegos de Datos

```
//Datos correctos AA.
$juego_datos_1 = array(
    'nombre' => 'Actividad 1',
    'id_nivel' => '1',
    'id_contexto' => '1',
    'tiempo_total' => '60'
);
//Datos parcialmente correctos AA (La actividad no tiene nombre).
$juego_datos_2 = array(
    'nombre' => ' ',
    'id_nivel' => '1',
    'id_contexto' => '1',
    'tiempo_total' => '60'
);
```

Descripción del Caso de Uso Modificar Actividad

Permite Modificar una actividad

```
public function Modificar($actividad) {
    if ($this->obj->ok($actividad['id_actividad'] != ' ', 'El parámetro nombre está correcto.')) {
      if($this->obj->ok($this->existeActividad($actividad['id_actividad']) != false, 'La actividad existe')) {
        if ($this->obj->ok($actividad['nombre'] != ' ', 'El parámetro nombre está correcto.')) {
          if ($this->obj->ok($actividad['id_nivel'] != ' ', 'El parámetro descripción está correcto.')) {
```

```
if($this->obj->ok($this->existeNivel($actividad['id_nivel']) != false, 'El nivel existe')) {
  if ($this->obj->ok($actividad['tiempo_total'] != ' ', 'El parámetro descripción está correcto.')) {
    if ($this->obj->ok($actividad['id_contexto'] != ' ', 'El parámetro descripción está correcto.')) {
      if($this->obj->ok($this->existeContexto($actividad['id_contexto']) != false, 'El contexto existe')) {
        $act = $this->existeActividad($actividad['id_actividad']);
        $act->setNombre($actividad['nombre']);
        $act->setLidNivel($actividad['id_nivel']);
        $act->setIdContexto($actividad['id_contexto']);
        $act->setTiempoTotal($actividad['tiempo_total']);
        $act->save();
        $this->obj->diag('La actividad fue modificada');
      } } } } }} }}}
```

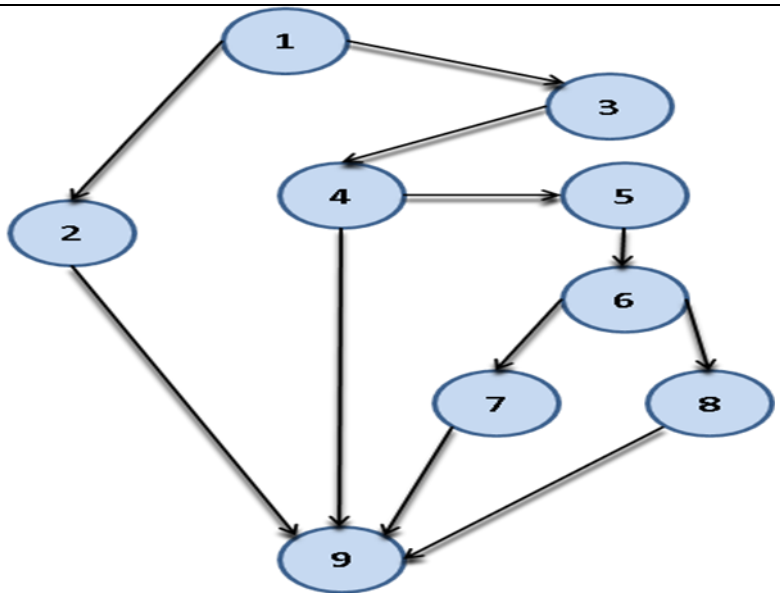
Complejidad ciclomática V (G) = 4	
Posibles Caminos 1-2-9 1-3-4-9 1-3-4-5-6-7-9 1-3-4-6-6-8-9	

Tabla 10: Escenario Modificar Actividad

Definir Juegos de Datos

```
//Datos correctos MA.
$juego_datos_10 = array(
   'id_actividad' => '6',
   'nombre' => 'Actividad 1',
   'id_nivel' => '2',
   'id_contexto' => '2',
   'tiempo_total' => '60'
```

```
);
//Datos incorrectos MA.
$juego_datos_10 = array(
    'id_actividad' => '6',
    'nombre' => ',
    'id_nivel' => '2',
    'id_contexto' => '2',
    'tiempo_total' => '60'
);
```

Descripción Caso de Uso Eliminar Actividad

Permite Eliminar una actividad

```
public function Eliminar($actividad) {
    if ($this->obj->ok($actividad['id_actividad'] != ' ', 'El parámetro id_actividad está correcto.')) {
      if($this->obj->ok($this->existeActividad($actividad['id_actividad']) != false, 'La actividad existe')) {
        TactividadPeer::eliminarActividad($actividad['id_actividad']);
      } } }
```

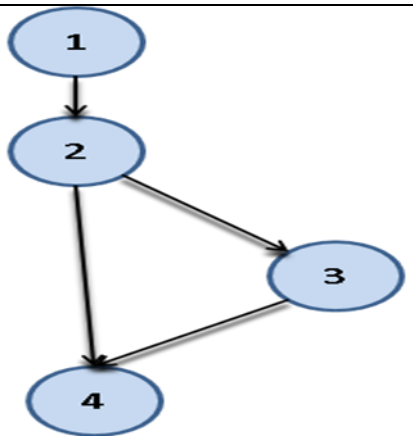
Complejidad ciclomática V (G) = 2	
Posibles Caminos 1-2-4 1-2-3-4	

Tabla 11: Escenario Eliminar Actividad

Definir Juegos de Datos

```
//Datos correctos EA.
$juego_datos_7 = array(
    'id_actividad' => '5'
);
//Datos parcialmente correctos EA (El id de la actividad no fue especificado).
```

```
$juego_datos_8 = array(
    'id_actividad' => ' '
);
```

3.5. Diseño de Casos de Prueba de Caja Negra aplicados

Diseño de Casos de prueba correspondiente al Caso de Uso: Adicionar Actividad

Descripción de la Funcionalidad:

El usuario entra los datos de la actividad a adicionar, en caso de existir una actividad con el identificador especificado el sistema muestra un error, de lo contrario queda adicionada la actividad.

Condiciones de Ejecución:

1 Se debe identificar y autenticar ante el sistema y además debe tener los permisos para ejecutar esta acción.
2 Se debe seleccionar el subsistema de Configuración.
3 Se debe seleccionar la opción Gestionar Actividad
4 Se debe seleccionar la opción Adicionar Actividad.

Secciones a Probar en el Caso de Uso

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Adicionar Actividad	EC 1.1: Adicionar una nueva Actividad.	El Caso de Uso comienza cuando el administrador accede a la interfaz Gestionar Actividad y oprime la opción Adicionar Actividad	1. El sistema muestra la interfaz que permite adicionar la Actividad, mostrando los campos que se deben llenar. 2. El administrador introduce los datos de la actividad y presiona la opción deseada. 3. El sistema verifica los datos introducidos por el administrador. 4. Si los datos introducidos son correctos, el sistema crea la actividad en la BD y termina el CUS.
	EC 1.2: Datos Incorrectos.	Consiste en detectar que faltan datos por llenar.	1. El sistema detecta que faltan datos por llenar y muestra un mensaje de error.

Tabla 12: Secciones a probar en el CU Adicionar Actividad

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Obs.
El usuario entra datos válidos para adicionar una actividad.		El sistema muestra un mensaje de éxito de la operación.	El sistema muestra el mensaje: "Actividad adicionada".	
	El usuario no especifica los datos de la actividad a adicionar.	El sistema muestra un mensaje de error.	El sistema muestra el mensaje: "Debe llenar todos los campos".	

Tabla 13: Caso de prueba Adicionar Actividad

Diseño de Casos de prueba correspondiente al Caso de Uso: Eliminar Actividad

Descripción de la Funcionalidad:

El usuario selecciona la actividad a eliminar especificando el identificador de la misma, en caso de no existir el sistema muestra un error, de lo contrario queda eliminada la actividad.

Condiciones de Ejecución:

1 Se debe identificar y autenticar ante el sistema y además debe tener los permisos para ejecutar esta acción.
2 Se debe seleccionar el subsistema de Configuración.
3 Se debe seleccionar la opción Gestionar Actividad
4 Se debe seleccionar la opción Eliminar Actividad.

Secciones a Probar en el Caso de Uso

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central

SC 1: Eliminar Actividad	EC 1.1: Eliminar Actividad	El Caso de Uso comienza cuando el administrador selecciona de la lista de actividades existentes la Actividad que desea eliminar.	1. El administrador selecciona la opción eliminar actividad. 2. El sistema muestra un listado con todas las actividades existentes en la BD. 3. El administrador selecciona la actividad que será eliminada y presiona la opción deseada 4. El sistema muestra un mensaje de advertencia para la acción a realizar. 5. El administrador confirma si desea o no eliminar la actividad seleccionada. 6. Si el administrador acepta, el sistema elimina la actividad seleccionada de la BD y termina el CUS.
	EC 1.2: Cancelar	Se cancela la acción	1. Si el administrador cancela la acción, se culmina el CUS sin ejecutar ninguna acción.

Tabla 14: Secciones a probar en el CU Eliminar Actividad

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Obs.
El usuario selecciona una actividad.		El sistema muestra un mensaje de éxito de la operación.	El sistema muestra el mensaje: "Se ha eliminado la actividad".	
	El usuario no selecciona una actividad.	El sistema muestra un mensaje de error.	El sistema muestra el mensaje: "Seleccione una actividad".	

Tabla 15: Caso de prueba Eliminar Actividad

Diseño de Casos de prueba correspondiente al Caso de Uso: Modificar Actividad

Descripción de la Funcionalidad:

El usuario selecciona la actividad, luego el sistema muestra una vista con los datos de la actividad, el usuario modifica los datos de la actividad, y si todos los datos nuevos son correctos queda modificada la actividad.

Condiciones de Ejecución:

1 Se debe identificar y autenticar ante el sistema y además debe tener los permisos para ejecutar esta acción.
2 Se debe seleccionar el subsistema de Configuración.
3 Se debe seleccionar la opción Gestionar Actividad
4 Se debe seleccionar la opción Modificar Actividad.

Secciones a Probar en el Caso de Uso

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Modificar Actividad	EC 1.1: Modificar Actividad.	El caso de uso comienza cuando el administrador selecciona la actividad que se desea modificar de la lista existente	1. El administrador selecciona la opción de Modificar actividad. 2. El sistema muestra todas las actividades existentes en la BD. 3. El administrador selecciona la actividad a modificar y presiona la opción Editar. 4. El sistema muestra la una interfaz con los datos anteriores de la Actividad y un formulario con los datos nuevos. 5. El administrador realiza los cambios necesarios a la actividad y presiona la opción deseada. 6. Si los datos están correctos el sistema modifica los datos de la actividad en la BD y termina el CUS.
	EC 1.2: Datos Incorrectos.	Consiste en detectar que faltan datos por llenar.	1. El sistema detecta que faltan datos por llenar y muestra un mensaje de error.

Tabla 16: Secciones a probar en el CU Modificar Actividad

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Obs.
El usuario entra datos válidos para modificar una actividad.		El sistema muestra un mensaje de éxito de la operación.	El sistema muestra el mensaje: “Se ha actualizado la actividad”.	
	El usuario no	El sistema muestra	El sistema muestra el	

	selecciona una actividad.	un mensaje de error.	mensaje: “ Seleccione una actividad”.	

Tabla 17: Caso de prueba Modificar Actividad

Diseño de Casos de prueba correspondiente al Caso de Uso: Visualizar Actividad

Descripción de la Funcionalidad:

El usuario selecciona la actividad a visualizar y el sistema muestra la actividad seleccionada.

Condiciones de Ejecución:

1 Se debe identificar y autenticar ante el sistema y además debe tener los permisos para ejecutar esta acción.
2 Se debe seleccionar el subsistema de Configuración.
3 Se debe seleccionar la opción Gestionar Actividad
4 Se debe seleccionar la opción Visualizar Actividad.

Secciones a Probar en el Caso de Uso

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC1:Visualizar Actividad	EC 1.1: Visualizar Actividad	El CUS se inicia cuando el usuario selecciona la actividad que desea seleccionar**.**	1. El usuario selecciona la actividad que desea visualizar 2. El sistema visualiza la actividad seleccionada 3. Termina el Caso de Uso.

Tabla 18: Secciones a probar en el CU Visualizar Actividad

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Obs.
El usuario selecciona una actividad.		El sistema muestra un mensaje de éxito de la operación.	El sistema muestra la actividad	
	El usuario no selecciona una actividad.	El sistema muestra un mensaje de error.	El sistema muestra el mensaje: “Seleccione una actividad”.	

Tabla 19 : Caso de prueba Visualizar Actividad

Diseño de Casos de prueba correspondiente al Caso de Uso: Realizar Actividad

Descripción de la Funcionalidad:

El usuario selecciona la actividad a realizar y el sistema muestra la actividad seleccionada.

Condiciones de Ejecución:

1 Se debe identificar y autenticar ante el sistema y además debe tener los permisos para ejecutar esta acción.
2 Se debe seleccionar el subsistema de Actividad.
3 Se debe seleccionar la opción Actividades a Resolver
4 Se debe seleccionar la opción Realizar Actividad.

Secciones a Probar en el Caso de Uso

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Realizar Actividad	EC 1.1: Realizar Actividad	El Caso de Uso comienza cuando el usuario selecciona la actividad que desee realizar	1. El usuario selecciona la actividad que desea realizar. 2. El sistema muestra un mensaje de advertencia para la acción a realizar. 3. El administrador confirma si desea o no eliminar la actividad seleccionada. 4. Si el usuario acepta, el sistema visualiza la actividad seleccionada de la BD y termina el CUS.
	EC 1.2: Cancelar	Se cancela la Acción	1. Si el usuario cancela la acción culmina el CU.

Tabla 20: Secciones a probar en el CU Realizar Actividad

Clases Válidas	Clases Inválidas	Resultado Esperado	Resultado de la Prueba	Obs.
El usuario selecciona una actividad.		El sistema muestra un mensaje de éxito de la operación.	El sistema muestra el mensaje: “Actividad realizada”.	

	El usuario no selecciona una actividad.	El sistema muestra un mensaje de error.	El sistema muestra el mensaje: “Seleccione una actividad”.	

Tabla 21: Caso de prueba Realizar Actividad

Después de realizarle un conjunto de pruebas al sistema se concluye que las pruebas constituyen un paso contundente en el ciclo de desarrollo del software que permiten verificar y revelar la calidad de un producto.

Para determinar el nivel de calidad se ha efectuado un conjunto de pruebas para comprobar el grado de cumplimiento respecto de las especificaciones iniciales del sistema. Lograr un software 100% libre de errores es un poco utópico aun más cuando se trata de sistemas complejos. De hecho, uno de los problemas más difíciles con la prueba es saber cuándo parar, debido en gran medida a la variedad de pruebas que pueden aplicarse a un sistema.

El capítulo se centró en la realización de Pruebas de Caja Blanca y de Caja Negra, para esto fueron definidos diferentes Casos de Prueba por cada Caso de Uso del Sistema, que permitieron detectar la ocurrencia de errores en la aplicación y la posterior solución de los mismos.

CONCLUSIONES

Se hizo un estudio de los antecedentes de los STI en el mundo, en Cuba y en la UCI, lo que permitió trazar una línea para el desarrollo de la aplicación.

Para implementar las funcionalidades se estudiaron los contenidos teóricos referentes a las características de las posibles herramientas a usar, lo cual permitió realizar una selección que se ajustara a las características del sistema a desarrollar. Se realizó un revisión bibliográfica de las tecnologías actuales para el desarrollo de sistemas web, enfatizando en las tecnologías y herramientas libres, de lo que se concluyó utilizar para la implementación del sistema, como lenguaje de programación PHP 5 en su versión 5.2.5, con el framework Symfony versión 1.2.8, sobre el entorno de desarrollo Netbeans 6.5 M2 para PHP y utilizando como SGBD PostgreSQL en su versión 8.3.4-1.

Además se validaron las funcionalidades en el producto para comprobar que cumpliera con los requisitos funcionales, lográndose una aplicación de fácil manejo, con una interfaz visual amigable y sencilla. La ejecución de las tareas en el sistema obtenido ocurre de forma aceptable.

Mediante el análisis de los resultados se llega a la conclusión de que se cumplió con el objetivo trazado en la investigación del presente trabajo de diploma.

RECOMENDACIONES

Con vistas a dar continuidad al desarrollo de este proyecto se recomienda:

- Implementar nuevas funcionalidades para mejorar la calidad y robustez del producto.
- Validar el funcionamiento de la herramienta a gran escala.
- Presentar este trabajo en futuros eventos científicos.

Referencias Bibliográficas

1. GestionProyectosPepe_Pino. [En línea] [Citado el: 15 de Febrero de 2010.] http://www.ciecem.uhu.es/centrodocumentacion/documentos/seminario/ponencias/GestionProyectosPepe_Pino.pdf.

2. [En línea] [Citado el: 20 de Diciembre de 2009.] http://www.gnuconsultores.com/es/ingenieria/desarrollo/escritorio.

3. **Bonaparte, Ing. Ubaldo.** *Paradigmas de Programación.* [En línea] 10 de Diciembre de 2004. [Citado el: 14 de Febrero de 2010.] http://www.frt.utn.edu.ar/sistemas/paradigmas/page22.html.

4. www.definición.org. [En línea] [Citado el: 18 de Diciembre de 2009.] http://www.definicion.org/lenguaje-de-programacion.

5. El lenguaje Java. [En línea] [Citado el: 9 de Diciembre de 2009.] http://www.dcc.uchile.cl/~lmateu/Java/Apuntes/java.htm.

6. www.PHP.net. [En línea] [Citado el: 21 de Diciembre de 2009.] http://www.php.net/manual/es/intro-whatcando.php.

7. CodeIgniter_Spanish_UserGuide. [En línea] [Citado el: 18 de Diciembre de 2009.] http://lax.franhp.net/CodeIgniter_Spanish_UserGuide.pdf.

8. **Potencier, Fabien y Zaninotto, François.** Symfony 1.2, La guía definitiva; Capítulo 1. Capítulo 1.

9. dojotoolkit.org. [En línea] [Citado el: 13 de Enero de 2010.] http://dojotoolkit.org.

10. extjses.com. [En línea] [Citado el: 13 de Enero de 2010.] http://extjses.com.

11. **Luciano.** Entornos de Desarrollo Integrado para Java. [En línea] [Citado el: 19 de Diciembre de 2009.] http://luauf.com/2008/05/13/entornos-de-desarrollo-integrado-para-java.

12. Zend.com. [En línea] [Citado el: 4 de Febrero de 2010.] http://www.zend.com/en/products/studio/features.

13. newcomers.php. [En línea] [Citado el: 2 de Febrero de 2010.] http://www.eclipse.org/home/newcomers.php.

14. [En línea] [Citado el: 2 de Febrero de 2010.] http://www.eclipse.org/pdt.

15. netbeans.org. [En línea] [Citado el: 14 de Enero de 2010.] http://www.netbeans.org.

16. MySQL. [En línea] [Citado el: 23 de Enero de 2010.] http://dev.mysql.com/doc/refman/5.0/es/index.html.

17. **Potencier, Fabien y Zaninotto, François.** Symfony 1.2, La guía definitiva; Capítulo 2. Capítulo 2.

18. www.angelfire.com. [En línea] [Citado el: 2 de 05 de 2010.] http://www.angelfire.com/empire2/ivansanes/bywbox.htm.

19. **Potencier, Fabien y Zaninotto, François.** Symfony 1.2, La guía definitiva; Capítulo 15. Capítulo 15.

20. **Giraffa, L. M. M.** *Selección y adopción de estrategias de educación en Sistemas Tutores Inteligentes.* Porto Alegre: CPGCC/UFRGS : s.n., 1997.

Bibliografía

Markiewicz, Marcus Eduardo y de Lucena, Carlos J.P. *El Desarrollo del Framework Orientado al Objeto.*

Bonaparte, Ing. Ubaldo. 2004. *Paradigmas de Programación.* [En línea] 10 de Diciembre de 2004. [Citado el: 14 de Febrero de 2010] http://www.frt.utn.edu.ar/sistemas/paradigmas/page22.html.

Cerami, Ethan. 2002. *Web Services Essentials.* s.l. : O'Reilly, 2002. 0-596-00224-6.

Collin, Peter. *Publishing Dictionary of Computing, 4th Edition.*

Cátedra de Paradigmas de Programación, Facultad Regional Buenos Aires, Universidad Tecnológica Nacional. *Fundamentos teóricos de los Paradigmas de Programación***.** [Citado el: 10 de febrero de 2010] http://www.pol.una.py/archivos/asi/paradigmas/FundamTeoricosParadigProg.pdf)

Vázquez, José Antonio Gallego. 2003. *Desarrollo Web con PHP y MySQL.* Madrid : ANAYA, 2003. 84-415-1525-5.

Welling, Luke y Thomson, Laura. *Programación Desarrollo Web con PHP y MySQL.* s.l. : ANAYA.

http://www.acm.org/crossroads/espanol/xrds7-4/frameworks.html

http://www.symfony.es/

http://www.symfony-project.org/more-with-symfony/1_4/es/

http://php.net/

http://www.postgresql.org/

http://www.postgresql-es.org/

http://codeigniter.com/

http://www.eclipse.org/

http://www.angelfire.com/empire2/ivansanes/bywbox.htm

Anexos

Anexo 1 Acta de Liberación

UCI

Acta de Liberación de Artefactos. Grupo de Calidad Centro CEGEL de la Facultad 10 de la Universidad de las Ciencias Informáticas.

Habana, 17 de junio de 2010

Luego de haber efectuado 2 iteraciones de pruebas al módulo Actividades del sistema Tutor virtual de evaluación para el aprendizaje autónomo de idiomas de la Facultad 10 y haberse detectado un promedio de 3 No Conformidades, se puede afirmar que se han corregido los defectos encontrados, por lo que queda liberada la aplicación.

Firma del Asesor y Jefe del Grupo de Calidad Centro CEGEL

Ing. Raúl Hernández Alvarez

UCI
Departamento Facultad 10

Ilustración 11: Acta de Liberación del sistema por parte del grupo de calidad.

Glosario de Términos

STI: Sistema Tutor Inteligente. Giraffa (1997) (19) los delimita como: “un sistema que incorpora técnicas de IA (Inteligencia Artificial) a fin de crear un ambiente que considere los diversos estilos cognitivos de los alumnos que utilizan el programa”.

HTML: HyperText Markup Language.

JavaScript: lenguaje de scripts, interpretado, multiplataforma y parcialmente orientado a objetos.

Ajax: Asynchronous JavaScript And XML.

API: Conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Herramienta: Aplicación empleada para la construcción de otros programas o aplicaciones.

IDE: Entorno de Desarrollo Integrado.

PEAR: Es un framework y sistema de distribución de compones reutilizables de PHP.

Plataforma: sistema operativo o sistemas complejos, ya sea de hardware o software, sobre el cual un programa pueda ejecutarse. Ejemplos típicos incluyen: arquitectura de hardware, lenguajes de programación y sus librerías de tiempo de ejecución. Ejemplos de plataformas son PC (Windows) y Macintosh (Mac).

Smaltalk: Primer sistema puro de objetos. Todo en Smaltalk es un objeto y toda la computación es desarrollada mediante mensajes que son enviados entre los objetos.

BD: Base de Datos.