Universidad de las Ciencias Informáticas
Facultad 2

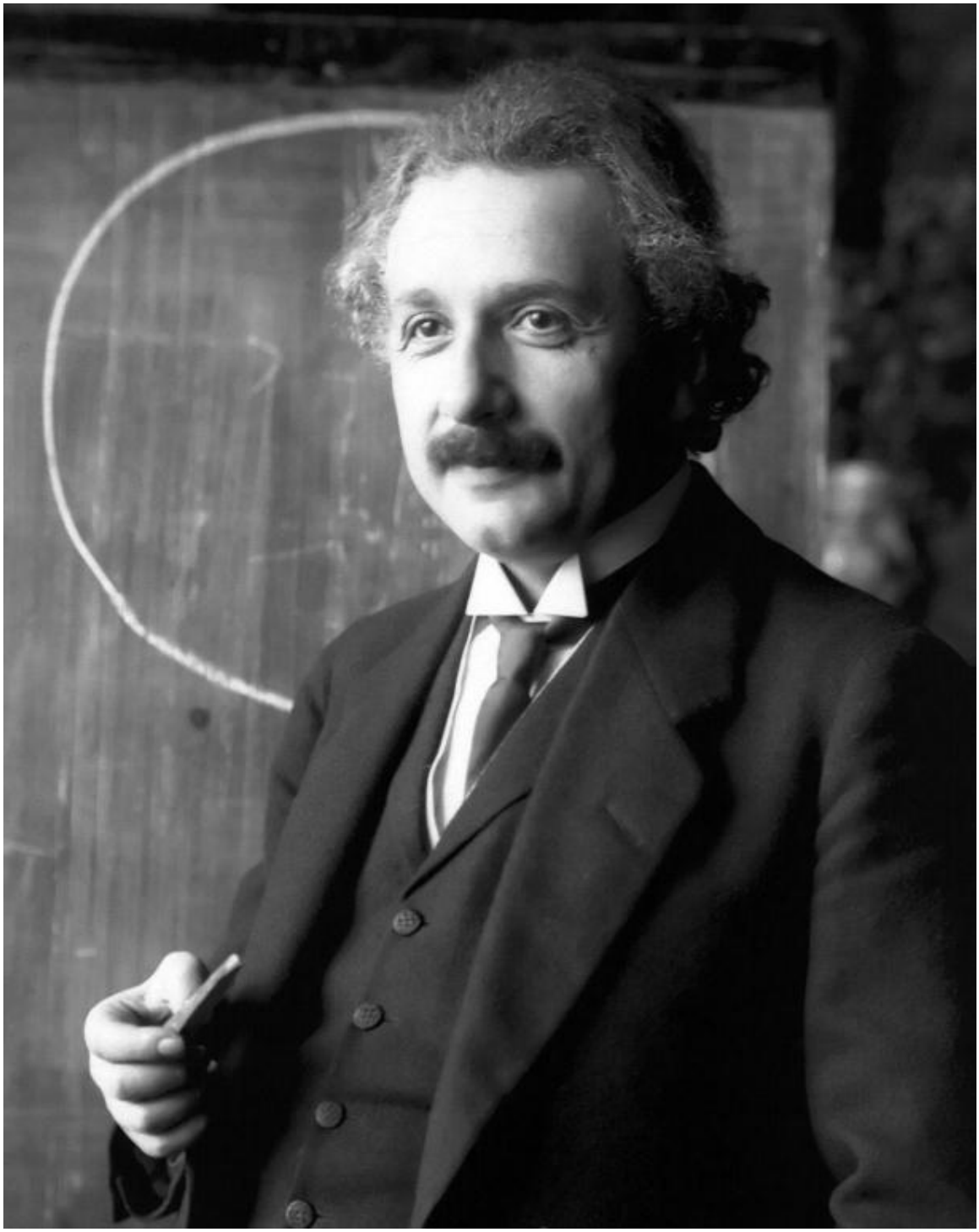
Sistema para la configuración de exámenes realizados en los laboratorios.

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas.

Autor(es): Olivio Marrero Vega

Tutor(es): Msc. Yahima Vigo Valdés

2013



Si buscas resultados distintos, no hagas siempre lo mismo.

Albert Einstein.

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas (UCI) a que haga el uso que estimen pertinente con el mismo.

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

Olivio Marrero Vega

Msc. Yahima Vigo Valdés

_____ _____

Firma del autor

Firma de la tutora

RESUMEN

En la Universidad de las Ciencias Informáticas (UCI), como en otros institutos educacionales, es necesario realizar exámenes comprobatorios. Los exámenes se realizan en formato duro o digital, dependiendo de las características de la asignatura que se evalúe. Para que este proceso se desarrolle de forma correcta se hace necesario el uso de locales previamente habilitados, con las tecnologías y los medios necesarios, como es el caso de los laboratorios. En dichos locales se comparten archivos y se determina el tiempo de realización del examen. También se determina desde dónde va a ser consultado el examen para garantizar la seguridad y minimizar la ocurrencia del fraude académico.

Actualmente en la universidad no existe una aplicación que permita a un profesor crear sus exámenes, configurarlos, guardar un histórico de los exámenes realizados anteriormente, calificar los exámenes y llevar un control de solicitudes hechas por estudiantes para la descarga de sus exámenes del historial. Dichas acciones son administradas a nivel central haciendo uso de aplicaciones que no poseen totalmente las funcionalidades para dar solución a estos problemas. Además no son administradas por profesores de los departamentos.

Tras realizar un análisis y estudio del funcionamiento de aplicaciones utilizadas para este fin, además de valorar el uso de estos medios, y comparar sus características y funcionalidades con las requeridas para el desarrollo de la aplicación, se decidió desarrollar un sistema para la configuración de los exámenes que se realizan en los laboratorios.

Como guía en el proceso de desarrollo de software se seleccionó la metodología RUP, y para la implementación del sistema se utilizó el lenguaje de programación PHP como lenguaje del lado del servidor, además de utilizar HTML y JavaScript como lenguajes de programación del lado del cliente, y JQuery y Symfony2 como Framework de desarrollo del lado del cliente y servidor respectivamente. Además se empleó el Gestor de Bases de Datos MySQL y el IDE NetBeans en su versión 7.3 como entorno de desarrollo.

Palabras Claves: Exámenes comprobatorios, Fraude académico, configuración de los exámenes.

Índice

Introducción

Tras años de desarrollo y evolución el ser humano ha aprendido a subsistir y dar solución a problemas de diversas índoles. En el transcurso de su desarrollo ha encontrado formas de realizar actividades que faciliten su existencia y actuación en el medio que lo rodea.

Actualmente muchas de las tareas que antes eran realizadas de forma manual, se automatizan, haciendo uso de modernas herramientas como son las computadoras. Dicha herramienta ha jugado y juega un papel fundamental en la vida del hombre, ordenando las actividades y tareas a realizar.

Una de las actividades comúnmente realizada por el hombre es la superación, que se desarrolla desde tiempos antiguos como un proceso de transformación y desarrollo a través del cual se trata de adoptar una nueva forma de pensamiento (es la acción y efecto de superar) (1). En la actualidad hay muchas formas de superarse, la más común y utilizada es llevada a cabo en institutos encargados de realizar estas actividades, donde profesores capacitados en diversas materias imparten clases, trasmitiendo sus enseñanzas a los alumnos que la reciben. Dichos institutos realizan exámenes periódicamente, como una forma de comprobar si los contenidos impartidos fueron aprendidos correctamente. De esta forma se realiza un proceso, en el cual el profesor encargado de realizar los exámenes debe manejar cuidadosa y correctamente para que no exista ningún contratiempo o problema en la realización del mismo.

No obstante, todos los días se busca mejorar y agilizar algunas de las actividades a desarrollar, utilizando nuevas herramientas y tecnologías en conjunto con las buenas prácticas, y remplazando las obsoletas. De esta forma aparecen nuevas alternativas de superación, proveyendo similares y mejores resultados, a un menor costo de recursos y materiales, así como esfuerzos por parte del personal encargado de realizar los exámenes comprobatorios.

Actualmente existen formas de capacitar y comprobar los conocimientos, estando el profesor y el alumno en distintas partes del mundo; esto sucede gracias al internet. Esta vía es muy utilizada por distintas empresas como una forma de capacitar al personal de su organización. También se realizan cursos online, los cuales una vez cursados certifican de forma digital que se posee los conocimientos adquiridos en el mismo (2). En este último escenario se brinda un contenido específico con determinadas características, las cuales proveen los conocimientos necesarios, evitando retraso y actualizando al cursante en poco tiempo. En la mayoría de estos cursos se ofrece una

fuente de teoría, videos explicativos así como laboratorios prácticos en los cuales se reafirma lo aprendido (3).

De este modo surge el término Educación a Distancia, el cual es una de las vías de superación más utilizadas por personas que comparten conocimientos y que se encuentran distantes geográficamente, poseyendo también los recursos o medios necesarios para realizar dicha actividad (4). El término Educación a Distancia se compone por dos palabras claves que separadas forman términos independientes.

El término educación se define como “La acción o proceso de educar o ser educado”, o también y de forma sencilla, “la acción de impartir conocimientos”. De estas definiciones se concluye que Educación a Distancia es una forma de educación, la cual es vista como un proceso realizado a distancia. Definiéndose como la acción o proceso de educar o ser educado, cuando este proceso se realiza a distancia (5).

El término Educación a Distancia ha sido definido por determinadas instituciones como:

- Una variedad de modelos de educación que tienen en común la separación física de los maestros y algunos o todos los estudiantes (6).
- La Educación a Distancia se realiza cuando los estudiantes y maestros están separados por la distancia física y la tecnología (voz, video, datos e impresiones) a menudo en combinación con clases cara a cara, es usada como puente para reducir esta barrera (7).

En estos escenarios se evidencia como el uso de las nuevas tecnologías puede influir en la vida del hombre, facilitando y mejorando en ocasiones el trabajo que se deba realizar.

En la UCI (Universidad de las Ciencias Informáticas) se realizan exámenes comprobatorios, los cuales siguiendo las características y medios que presenta la universidad pueden ser en formato duro o digital, o ambos inclusive. Dichos exámenes son realizados tanto en las aulas como en los laboratorios. En caso de que estos exámenes se realicen en los laboratorios existen ocasiones en que su distribución y recogida se realiza de forma manual, algunas veces con dispositivos extraíbles provocando retrasos en el comienzo de los mismos, además de que algunas computadoras presentan problemas en la lectura de dichos dispositivos.

Otra manera de realizar el proceso de distribución y recogida de exámenes es compartiendo una carpeta en una computadora, que puede ser del profesor o del técnico del laboratorio, con la información necesaria para la realización del examen. Esta variante puede tener el inconveniente del acceso concurrente al mismo lugar donde están los exámenes del resto de los estudiantes. Además de que propicia al

fraude, y puede presentar problemas en las conexiones, pues las computadoras tienen un número límite de conexiones concurrentes impidiendo que algunos estudiantes puedan acceder a los recursos compartidos. También se pueden perder exámenes, y entregar ficheros no modificados o en blanco sin previo aviso o alerta de lo ocurrido, y no se lleva un control de la cantidad de exámenes subidos con respecto a la cantidad de estudiantes del aula. Para realizar este proceso el profesor responsable de cuidar el examen debe realizar el conteo manual de los ficheros en el recurso compartido.

Para entregar los ficheros del examen también se emplea en algunas ocasiones como alternativa, el correo electrónico de la universidad, que el hecho de permitir que se abra el mismo en plena realización del examen puede conllevar al fraude, además que se realiza un uso indebido de las tecnologías de la universidad. Esta vía tampoco es factible ya que se pueden enviar ficheros no modificados o en blanco subidos accidentalmente.

Otra variante de realización de exámenes es la plataforma Moodle empleada en la universidad, que permite publicar toda la información necesaria de un examen, así como subir los ficheros resultantes de la realización del mismo. En esta herramienta no se pueden establecer configuraciones de exámenes donde se tenga en cuenta aplicarla a un año, facultad, o grupo, los cuales pueden ser de diversas facultades para el caso de los exámenes extraordinarios, además de que exige una dependencia del asesor provocando retrasos en ocasiones. También presenta la deficiencia de no llevar un histórico, del cual se puedan descargar los exámenes realizados anteriormente. Esta última característica en caso de ser aplicada, estaría determinada por el profesor responsable de la asignatura, el cual sería el encargado de otorgar los permisos necesarios para que esto suceda. Otra deficiencia que presenta son sus múltiples usos, ya que se encuentra no solo el examen a aplicar, sino también todo el contenido de las asignaturas que recibe el estudiante, pudiendo propiciar el fraude académico. Además de que no posee un mecanismo de aviso que alerte al estudiante cuando esté subiendo un fichero no modificado o vacío.

Dada las situaciones antes expuestas surge el siguiente **problema a resolver**: ¿Cómo efectuar el proceso de realización de los exámenes en los laboratorios de manera que permita la configuración de los exámenes para evitar y prevenir fraudes académicos y agilizar dicho proceso?

El **objeto de estudio** es: El proceso de realización de exámenes en los laboratorios. Se plantea como **objetivo general** de la investigación: Desarrollar una aplicación informática que permita agilizar el proceso de realización de exámenes en los laboratorios y contribuya a evitar y prevenir fraudes académicos.

El **campo de acción** lo constituye: El funcionamiento de las distintas aplicaciones que permiten configurar exámenes para determinadas asignaturas y cursos online.

Las **preguntas científicas** que guiarán la investigación, de acuerdo con el problema científico y las necesidades detectadas, son:

- ¿Cómo se efectúa el proceso de realización de los exámenes en los laboratorios?
- ¿Cuál es el estado del arte en las distintas aplicaciones que permiten la configuración de archivos de exámenes para determinadas asignaturas o cursos?
- ¿Qué tecnologías y herramientas emplear para lograr la correcta configuración de los exámenes en la universidad, así como para agilizar el proceso de realización de los mismos?

Como resultado de la relación entre el problema científico y el objeto de estudio se desarrollaron los siguientes **objetivos específicos**:

- Definir el proceso de realización de exámenes en la Universidad de las Ciencias Informáticas, teniendo en cuenta todos los tipos de exámenes de las diferentes asignaturas.
- Realizar un estudio del proceso de realización de exámenes en la universidad, así como de las herramientas utilizadas en el mismo.
- Diseñar la aplicación que permita gestionar el proceso de realización de exámenes y la configuración de los mismos.
- Implementar la aplicación informática que se encarga de gestionar el proceso de realización de exámenes y configuración de los mismos, permitiendo agilizar dicho proceso.
- Validar la solución propuesta.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En el siguiente capítulo se desarrolla un estudio del proceso de realización, distribución y recogida de los exámenes realizados en los laboratorios de la universidad. También se mencionan y caracterizan aplicaciones relacionadas con el proceso, que aunque no lo efectúan de igual manera, o completamente, poseen componentes que pueden servir de guía, ejemplo o ayuda para la aplicación a desarrollar. Se especifican las pautas a seguir para que este proceso se desarrolle correctamente, así como los requisitos con los que se debe cumplir para que se logre mejorar o perfeccionar dicho proceso. Además se hace un análisis de las herramientas y medios informáticos necesarios y convenientes para desarrollar la aplicación que se encargará de realizarlo.

1.1. Proceso de realización, distribución y recogida de exámenes en los laboratorios

El proceso de realización del examen comienza desde que el profesor encargado del grupo que se dispone a realizar el examen, en conjunto con el mismo entran al laboratorio, preparándose y revisando la tecnología, esperando hasta la hora exacta de comienzo para la revisión y lectura del archivo del examen. En ocasiones y dependiendo de la asignatura de la cual se realiza el examen, los archivos son entregados y recogidos de forma manual, o con el apoyo de diversas aplicaciones informáticas, agilizando de cierta manera dicho proceso.

La vía manual se desarrolla en caso en que el laboratorio escogido no presente conexión de red. También en caso en que la matrícula del grupo exceda la capacidad de puestos disponibles del laboratorio y sea necesario trasladar los alumnos sobrantes a otro local en los que la tecnología no esté preparada para el examen. Puede darse el caso también que la asignatura precise ser evaluada en formato duro y digital al mismo tiempo. En dicha evaluación primero se prepara y desarrolla la parte dura, dando tiempo a que se despliegue por los puestos los archivos de la parte digital, utilizándose para esto un dispositivo extraíble, el cual es el medio más común y rápido en estas circunstancias.

Por otra parte la vía en la que se utilizan los medios informáticos como ayuda es la más común y utilizada. Suele ocurrir en circunstancias normales, en las que no existan problemas con la red en los laboratorios, y la tecnología de los mismos no se encuentre defectuosa, encontrándose lista y disponible para la realización del examen. Esta vía es utilizada también en el caso en que el examen precise ser evaluado de forma digital y dura, haciendo uso de aplicaciones informáticas para publicar los archivos del examen de la parte digital, y para que los estudiantes procedan a su descarga y revisión desde

la misma. Además ofrece la ventaja de ahorrar tiempo en la preparación y realización del examen, así como en su distribución y recogida, ya que no se realiza de forma manual. Mayormente la aplicación que se utiliza para estos fines es la plataforma Moodle, la cual presenta disímiles características, entre ellas, publicar y permitir la subida de los ficheros del examen a la misma.

Cuando los exámenes no son realizados a nivel central, y se realizan a nivel de facultad, el profesor encargado, como por ejemplo el de programación de la facultad 2, presenta una pequeña aplicación en la que se publica el examen. Dicha aplicación puede ser desplegada en el mismo laboratorio que se efectúe el examen, en la computadora del técnico, o en la propia computadora del profesor situada en su departamento.

Otra forma de facilitar el proceso de distribución y recogida de los exámenes es compartiendo una carpeta en la computadora del técnico o el profesor al frente del laboratorio, posibilitando así que los estudiantes accedan a la misma para la obtención y entrega del examen.

También pudiera suceder que se utilice el correo electrónico como medio auxiliar para la entrega de los ficheros resultantes del examen, en caso de que algunas de las variantes anteriores fallaran o no fueran las propicias. Esta última variante es la menos utilizada al presentar cierto inconveniente al profesor, ya que le llena su buzón de correo, pudiéndose extraviar o confundir algún examen. Además de que pudiera ser utilizada para el fraude académico por parte de los estudiantes.

A partir de un estudio de las anteriores variantes de realización, distribución y recogida de exámenes, se hace necesaria la existencia de un medio que garantice agilizar el proceso de realización, distribución y recogida de exámenes. También que permita disminuir la posibilidad de fraude, además de permitir a los estudiantes conocer si se ha enviado un fichero no modificado o vacío, el conteo de archivos por estudiantes, brindar datos estadísticos, entre otras funciones.

1.2. Aplicaciones relacionadas con el proceso de realización, distribución y recogida de exámenes.

1.2.1. Plataforma Moodle

Moodle es un Sistema de Control y Administración de Contenidos especializado en educación a distancia (8), que permite la creación de sistemas de enseñanzas completos en línea. Es una de las plataformas más usadas a nivel educativo (9). Crea diversidad de actividades como cuestionarios, encuestas, glosarios de tipo abierto, adjuntar archivos, blogs, wikis, *hot potatoes*, entre otras.

Tiene como característica fundamental que se encuentra montada sobre la Web, poseyendo como interfaz gráfica una página Web, haciéndola asequible desde distintos sistemas operativos. Además de que se distribuye gratuitamente como software libre bajo la licencia pública GNU GPL, del inglés *General Public License*.

Moodle soporta varios tipos de base de datos, entre ellas la más sugerida MySQL. Usa la librería ADOdb para la abstracción de bases de datos. Es una plataforma fácil de instalar, aprender y modificar, y fácil de actualizar desde una versión a la siguiente. Presenta varias características modulares lo que permite a cualquiera añadir características al código básico principal o incluso distribuirlas por separado. Además de que puede integrar una plataforma con otra que se encuentre en la red, formando así una gran comunidad de sistemas de enseñanzas a distancia. También mantiene todos los archivos para un curso en un único directorio en el servidor.

Actualmente en la UCI se despliega una plataforma Moodle, a la cual todos los estudiantes y profesores del centro la conocen como EVA (Entorno Virtual de Aprendizaje). La misma es utilizada como un espacio de apoyo al proceso de formación de la Carrera de Ingeniería en Ciencias Informáticas, donde los profesores pueden implementar estrategias de enseñanza-aprendizaje complementarias a las clases presenciales, así como diseñar cursos semipresenciales o totalmente a distancia, proveyendo a los estudiantes de un poderoso medio en el cual pueden obtener, utilizar o compartir materiales didácticos (10).

Entre sus desventajas se puede enunciar que algunas actividades pueden ser realizadas de forma mecánica, dependiendo en gran parte del diseño de las instrucciones. Al estar basado en tecnología PHP, la configuración de un servidor con muchos usuarios debe ser cuidadosa para obtener de esta forma el mejor desempeño. No posee un mecanismo que permita automatizar el cambio de estado a visible para los estudiantes, para determinado día, a una determinada hora, un examen ya publicado. No lleva un histórico de los exámenes que se suben a dicha aplicación. No posee un mecanismo de alerta, que avise cuando un fichero es entregado sin modificar o en blanco. Además de que no se configuran exámenes para un año, facultad o grupo, solo se publican exámenes a nivel central.

1.2.2. Herramienta Claroline

Claroline es un conjunto de software (*groupware)* asíncrono y colaborativo (11). Es una herramienta estable de código abierto, para la creación de cursos online. Se desempeña bajo la licencia GPL (*General Public License*), y actualmente se encuentra desplegada en más de 100 países. Está escrita en el lenguaje de programación PHP y utiliza MySQL como SGBD (Sistema Gestor de Base de Datos). Presenta las

características propias de un CMS (Sistema de Gestión de Contenidos), además de ser utilizado por formadores, para administrar cursos virtuales en entornos de aprendizaje en línea (*e-learning* en inglés). Permite a los profesores insertar la descripción de un curso, publicar documentos en cualquier formato, administrar foros de discusión públicos y privados, estructurar una agenda con tareas y plazos, publicar eventos, gestionar wiki, comprobar las estadísticas de los ejercicios y de asistencia, preparar ejercicios online, hacer anuncios vía correo electrónico y crear y guardar chats (12).

A diferencia de otras plataformas como Moodle, Claroline no presenta abstracción de la base de datos y depende totalmente de MySQL. Es una herramienta poco modificable y tampoco permite exportar los cursos que ofrece. Además de que cuenta con pocos módulos y *plugins* para descargar, y posee dificultad para su personalización (13). Por otro lado en similitud al Moodle no presenta mecanismos que generen alertas en caso de subida de un fichero no modificado o en blanco, tampoco permite la configuración de descargas de archivos de exámenes para un año, facultad o grupo, y no guarda un historial de exámenes realizados anteriormente.

1.2.3. Herramienta GESPRO

La herramienta GESPRO es un paquete para la gestión de proyectos desarrollado por la Universidad de las Ciencias Informáticas (UCI) y comercializable desde las empresas comercializadoras asociadas a la universidad. Su desarrollo se basó en Redmine, la cual es una herramienta de gestión de proyecto y seguimiento de errores que posee una interfaz web (14). Se encuentra actualmente desplegada en la universidad, y surge tras la necesidad de la búsqueda de nuevas formas de organización y control de las producciones. Se encarga de organizar y administrar recursos de manera tal que se pueda culminar todo el trabajo requerido en los proyectos dentro del alcance, el tiempo, y coste definidos. Dicha herramienta se desarrolló con el fin de facilitar el trabajo de los especialistas, proveyendo comodidades y medios para realizar acciones que antes se realizaban de forma manual (15).

Aunque no presenta relación directa con los exámenes realizados en la universidad, esta herramienta permite de igual forma la configuración de diferentes acciones. Estas acciones pueden ser, la asignación de tareas, subida a la plataforma del fichero resultante de la tarea, actualización del estado de la tarea, entre otras. Realizando así un proceso similar al que se estudia.

Dicha herramienta permite que un profesor perteneciente a determinado proyecto con los permisos necesarios pueda ser capaz de asignar una tarea a un estudiante específico en su proyecto. Además posee un subsistema de correo y mensajería que permite las notificaciones al usuario ante determinados cambios, como por ejemplo

cuando una nueva tarea es asignada, es actualizada, o cerrada. La tarea asignada posee un lapso de tiempo dentro del cual debe ser resuelta, y un estado, el cual el estudiante al que se le asigna debe cambiar a medida que realiza la misma. Una vez realizada dicha tarea el estudiante debe subir el fichero resultante a la aplicación y cambiar su estado a resuelta.

También posee la seguridad y el acceso controlado necesario como para no permitir que colapse el sistema por la realización de alguna acción malintencionada, identificando las personas que acceden a la misma. La interacción con la herramienta es a través de la Web, por lo que es necesario tener instalado un navegador web como, Mozilla, Opera, Internet Explorer entre otros. Además posee herramientas para la generación dinámica de reportes y el análisis estadístico, así como también para el trabajo colaborativo como foros, wiki, vigilancia tecnológica, y meta buscadores.

Esta herramienta enfoca sus procesos en la gestión de proyectos, y aunque permite configurar y realizar acciones, como la asignación de tareas y subida de archivos a la misma. No permite publicar ficheros y tampoco configurarlos para que sean expuestos para un año, facultad o grupo de distintas facultades. Además que tampoco genera una alerta en caso de que se suba un fichero no modificado o vacío a la plataforma.

1.2.4. Plataforma Blackboard

Blackboard es una plataforma computacional para la administración del aprendizaje en línea (*e-learning*). Entre sus características se puede enunciar que es flexible, sencilla y de uso intuitivo, además integra un ambiente sólido de enseñanza y aprendizaje en línea. Se diseñó para ser utilizada por instituciones dedicadas a la enseñanza y el aprendizaje, y proporciona la funcionalidad necesaria para administrar correctamente programas de educación a distancia o a través de la Web (16).

Ofrece herramientas de creación de cursos y contenidos, una nueva manera de evaluación, herramientas de colaboración síncronas y asíncronas, así como la administración académica de los estudiantes por parte de profesores. Permite administrar un conjunto de recursos que posibilita desarrollar cursos virtuales, dentro de estos específicamente, impartir y distribuir contenidos que se encuentran presentados en diversos formatos, como texto, sonido, video y animación. También realizar evaluaciones en línea que se califican de forma instantánea, llevar a cabo el seguimiento académico de los alumnos participantes, asignar tareas y desarrollar actividades en ambientes colaborativos.

Dicha herramienta posee varias limitaciones, entre las cuales se pueden enunciar que el acceso a sus cursos debe ser en línea, utilizando un navegador de internet, ya que no

posee la opción de obtener una versión local del curso. Es un software privativo que tiene un costo, lo que no lo hace accesible a cualquier usuario, además que algunas definiciones se deben hacer en código HTML (*HyperText Markup Language* de sus siglas en ingles), lo que impone la necesidad de conocer los detalles básicos sobre el mismo (17).

1.2.5. Plataforma Sakai

Sakai es una plataforma en línea para el manejo de cursos, al igual que Blackboard y Moodle. Es un LMS (*Learning Management System* de sus siglas en ingles), que dispone de un ambiente de aprendizaje y colaboración que añade herramientas para el desarrollo de contenidos, coordinación de proyectos y gestión de portafolios. Se desarrolló con tecnología Java, proporcionándole una gran flexibilidad y permitiéndole ser utilizada en organizaciones con todo tipo de requisitos (18).

Dicha plataforma es un software con licencia gratuita y código abierto, lo que le permite ser ampliado y mejorado para cubrir las necesidades de las instituciones que la utilicen. Ofrece una amplia lista de herramientas que abarcan en su mayoría las necesidades de formación *on-line* de una organización, además permite brindar apoyo a la formación presencial, semipresencial y completamente a distancia. También se usa para crear espacios de trabajo colaborativo, prácticos para investigación, o para el desarrollo de proyectos.

Entre sus principales funcionalidades Sakai permite tener un sitio web diferenciado para cada curso con una interfaz común y un acceso único. Genera guías didácticas y crea anuncios del curso. Mantiene una agenda actualizada para cada una de las materias. Realiza pruebas de evaluación o encuestas a los usuarios. Realiza trabajos colaborativos o individuales. Genera y modera discusiones en foros y otras herramientas de comunicación. Envía mensajes privados a los usuarios. Presenta contenidos basados en las nuevas tecnologías educativas. Crea blogs, wikis, distribuye *podcasts* y otros recursos multimedia. Gestiona los roles y permisos de los usuarios. Realiza el seguimiento del progreso y estadísticas de los alumnos. Permite que los profesores realicen su trabajo en un entorno común. Ofrece la opción de personalizar el aspecto visual para coincidir con la imagen corporativa de la organización que la adquiera. Proporciona una interfaz flexible y adaptable a cada uno de los cursos que se imparten. Permite cargar y descargar archivos en el servidor del Campus Virtual, los cuales podrá mantener como privados o hacerlos disponibles a los alumnos que los requieran. Además de poder integrarse con los sistemas propios de la organización.

1.2.6. Justificación de la necesidad de desarrollar una aplicación informática.

Tras el estudio del funcionamiento y las características de las herramientas anteriores, se puede determinar que la aplicación a desarrollar debe poseer un sistema de seguridad que permita la autorización y el acceso restringido a determinadas funcionalidades, de acuerdo a los permisos otorgados en la misma, siguiendo la estructura de la herramienta GESPRO. Además no debe crear dependencias como sucede en el caso de la plataforma Moodle, en la cual se hace necesaria la colaboración de un asesor para gestionar determinada actividad que el profesor necesite realizar. También sería necesario llevar un histórico de pruebas realizadas anteriormente, que facilite la descarga de las mismas tras tener los permisos necesarios otorgados por el profesor de la asignatura comprendida. Esta característica no se pone de manifiesto en la plataforma Moodle ni en la herramienta Claroline.

Otra característica esencial en el tema de la seguridad y relacionada al tema de los fraudes, sería el control de las trazas dejadas tras la navegación del sitio, proporcionando información detallada de los lugares en los que se estuvo, los documentos descargados, así como las fechas de creación y modificación de los documentos. Dichas características tampoco las presentan las plataformas y herramientas estudiadas anteriormente.

En las herramientas anteriores no se menciona el manejo del estado de los exámenes, permitiendo con esto que solamente se suba un solo examen por usuario. También adjunto a dicho tema se encuentra la configuración de los exámenes, en el caso de que se necesiten establecer todos los requisitos como por ejemplo la dirección IP desde la que se permite subir el examen, el encargado de subirlo, y la bibliografía anexa.

Otras de las características más importantes que se necesita y que no presentan estas herramientas es la de permitir publicar un examen automáticamente. Esto sería necesario para el caso que se decidiera publicar un examen un día determinado a una hora determinada, permitiendo configurar esta opción.

Además se demuestra la necesidad de seguir una de las características presentadas en las herramientas anteriores, en la cual se maneja un sistema de mensajería y aviso que mayormente se destina al correo electrónico de la universidad para enviar notificaciones y emitir alertas.

1.3. Metodología de Desarrollo de Software a utilizar en el desarrollo de la aplicación.

RUP (*Rational Unified Process*) fue la metodología que se decidió utilizar, ya que la aplicación que se quiere desarrollar será de utilidad en la universidad como herramienta

de apoyo, en el proceso de realización, distribución y recogida de exámenes, lo que permitirá que se modifique o amplíe de acuerdo a las necesidades del instituto. Su posterior modificación o ampliación se podrá llevar a cabo gracias a la extensa documentación y cantidad de artefactos que genera la metodología seleccionada.

Dicha metodología se enfoca en gran medida a las características que presenta el producto que se quiere obtener, como por ejemplo ser modelado y diseñado por casos de uso, centrándose en la documentación, planificación y procesos, los cuales son aspectos esenciales en el desarrollo del mismo. Es una metodología robusta que permite el desarrollo de software a gran escala, mediante un proceso continuo de pruebas y retroalimentación, garantizando el cumplimiento de ciertos estándares de calidad. Se centra en asegurar que el producto satisfaga los requerimientos de los usuarios finales. Posee varias características que la hacen ideal para el desarrollo del software, entre ellas las empleadas en el desarrollo del sistema, entre las que se pueden encontrar que es guiado por casos de uso, centrado en la arquitectura, y utiliza UML *(Unified Modeling Language*) como lenguaje de notación. RUP es el proceso de desarrollo más general de los existentes actualmente, y proporciona muchas ventanas sobre metodologías ágiles como es XP, al dar énfasis en los requisitos y el diseño, presentando con esto aspectos esenciales para dar continuidad al desarrollo de la aplicación (19).

1.4. Herramienta CASE

La herramienta que se seleccionó fue el Visual Paradigm en su versión 8.0. La misma es una herramienta CASE (Ingeniería de Software Asistida por Computación (traducido al español)). Esta herramienta propicia un conjunto de ayuda para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Se escogió esta herramienta porque entre sus características más significativas se puede encontrar que trabaja bajo una licencia gratuita y comercial, se despliega sobre múltiples plataformas, como son, Linux y Windows, y soporta aplicaciones Web. Su diseño es centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad. También el modelo y el código permanecen sincronizados en todo el ciclo de desarrollo (20).

1.5. Lenguajes de programación web a utilizar

1.5.1. PHP (Hipertext Preprocesor)

El lenguaje de programación a utilizar del lado del servidor es PHP en su versión 5.4.3, debido a que es el lenguaje que se estableció, según el Framework de la aplicación.

Entre otros elementos se escogió PHP porque es un lenguaje de programación rápido, gratuito, e independiente de plataforma. Este lenguaje del lado del servidor permite la conexión con distintos gestores de bases de datos, y puede escribirse en pequeños fragmentos dentro del código HTML (*HyperText Markup Language*), lo que lo hace realmente fácil de utilizar. Es un lenguaje que ha tenido una gran aceptación en la comunidad de desarrolladores, debido a la potencia y simplicidad que lo caracterizan, así como al soporte generalizado en la mayoría de los servidores (21).

Además este lenguaje de programación es ideal para realizar diversos tipos de aplicaciones Web, gracias a la extensa librería de funciones con que cuenta. La librería de funciones posee múltiples usos, brindando desde cálculos matemáticos complejos, hasta tratamiento de conexiones de red.

1.5.2. HTML (HyperText Markup Language)

Como lenguaje de programación del lado del cliente se escogió HTML 5. El mismo es un lenguaje de marcado predominante en la elaboración de páginas Web. Se utiliza para describir y traducir la lectura y la información en formatos de textos, así como para complementar el texto con objetos tales como imágenes.

Los documentos HTML son ficheros de texto, que pueden ser modificados con cualquier editor de texto, como por ejemplo, el bloc de notas de Windows. El nombre de los ficheros escritos en lenguaje HTML suelen tener la extensión HTML o HTM (*HyperText Markup*). Este lenguaje está compuesto por etiquetas que definen la estructura y el formato del documento, que estará disponible para el usuario a través de la web. También posee varias ventajas, como por ejemplo, crear enlaces a distintas partes del documento, o a distintas fuentes de datos, a través de hipervínculos o hiperenlaces. HTML no es un lenguaje compilado, por lo que cualquier error de edición se mostrará en forma de texto plano, sin el formato adecuado (22).

1.5.3. JavaScript

JavaScript es un lenguaje de programación interpretado, el cual es utilizado para crear páginas Web dinámicas. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico (23).

Este lenguaje de programación es interpretado, por lo que no necesita ser compilado para probar los programas hechos con él. Es un lenguaje interpretado el cual se integra en los documentos HTML, esto es realizado mediante un enlace al archivo con extensión “.js” que contiene el código, o incrustándole directamente el código en el documento, haciendo uso de algunas etiquetas para dar a conocer al cliente o

navegador que ese documento HTML presenta código JavaScript. Además requiere estar activado en los navegadores para que su uso sea posible por estos.

1.6. Sistema Gestor de Bases de Datos escogido.

Se decidió la utilización del sistema de gestión de base de datos relacional y multiusuario MySQL en su versión 5.5.24 para la aplicación porque entre sus características más importantes se puede encontrar que es multiplataforma. Soporta grandes bases de datos y es sencillo y cómodo de usar. Además que posee una perfecta integración con aplicaciones escritas con el lenguaje de programación PHP. Es un sistema muy rápido en la lectura de base de datos cuando utiliza el motor no transaccional MyISAM, pudiendo provocar también problemas de integridad en entornos de alta concurrencia en la modificación de datos. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que lo hace ideal para este tipo de aplicaciones (24).

1.7. Servidor web a utilizar por la aplicación.

Se determinó la utilización de Apache 2.2.22 como servidor Web, ya que presenta numerosas características que lo sitúan entre los primeros de la lista de los servidores más utilizados en todo el mundo. Es un servidor Web de tecnología libre (*Open Source*), de sólida estructura y para uso comercial. Permite ser configurado y es altamente robusto, además de multiplataforma, siendo casi universal al ser utilizado por muchos Sistemas Operativos. Interactúa con una gran cantidad de lenguajes, como Perl, PHP, JavaScript, entre otros lenguajes de script, proveyendo también tecnologías para el desarrollo de páginas JSP, y brindando todo el soporte que se necesita para tener paginas dinámicas. Además posee un diseño modular, permitiendo ampliar sus capacidades (25). Por estas características y ventajas se decidió utilizar al servidor Web Apache para publicar la aplicación.

1.8. Entorno de Desarrollo Integrado (IDE) escogido para el desarrollo de la aplicación.

Se escogió como Entorno de Desarrollo Integrado (IDE), el NetBeans 7.3, ya que gestiona de manera sencilla y rápida los proyectos. Este IDE en comparación con otros posee compatibilidad no solo con PHP, sino con una gran cantidad de lenguajes para plataforma web como son, HTML, CSS, XML, JavaScript, SQL y Java. Es cómodo de usar y posee facilidades para el trabajo con proyectos como navegadores de proyectos, de soluciones, de archivos, entre otros. También posee integración con la mayoría de *Framework* actuales, y gestiona su visualización a través del explorador por defecto del sistema o el que se predetermine.

1.9. Framework para el desarrollo de la aplicación

Librería JavaScript escogida para el desarrollo de la aplicación

JQuery es una librería JavaScript que fue creada en el 2006 con el fin de resolver el problema de manejo de eventos, animaciones, Ajax y DOM**.** La misma fue escogida en su versión 2.0 para el desarrollo de la aplicación, ya que es una librería liviana que enfatiza la interacción entre JavaScript y HTML. Además de ser considerada la librería más utilizada en la actualidad para el desarrollo web, es al mismo tiempo utilizada por grandes corporaciones, entre las que se encuentra Microsoft, la cual integra jQuery en Visual Studio para el uso en aplicaciones desarrolladas en ASP.NET. Además Nokia también usa jQuery en sus aplicaciones web (26).

Framework PHP escogido para el desarrollo de la aplicación

El *Framework* PHP escogido para el desarrollo de la aplicación fue Symfony en su versión más reciente nombrada Symfony2. Este *Framework* facilita el desarrollo de las aplicaciones web. Se encarga de todos los aspectos comunes y más tediosos a desarrollar en dichas aplicaciones, enfocando al desarrollador a aportar valor a las características únicas de cada proyecto. Esta versión se anuncia por primera vez a principios del 2009, y supone un cambio radical tanto en arquitectura interna como en filosofía de trabajo respecto a sus versiones anteriores. Además Symfony2 fue ideado para llevar al límite todas las nuevas características de PHP 5.3, posicionándose como uno de los *Framework* PHP con mejor rendimiento, permitiendo además reemplazar o eliminar fácilmente aquellas partes que no encajan en un proyecto (27).

Conclusiones

Las herramientas utilizadas en el proceso de realización de exámenes en la universidad necesitan tener incorporadas un conjunto de funcionalidades y características que permitan desarrollar dicho proceso con eficacia y rapidez. Debido a que actualmente no se emplean herramientas completamente ideales para el desarrollo del proceso, conlleva a que se produzcan problemas de diversas índoles, entre ellos el fraude académico y en ocasiones un retraso en la recogida y distribución de los archivos del examen. En el desarrollo del actual trabajo se realizó un estudio del estado del arte de las herramientas informáticas relacionadas con el proceso descrito anteriormente, llegando a la conclusión de que no pueden ser usadas en el sistema a desarrollar. Por tal motivo es necesaria la implementación de un sistema para la configuración de exámenes realizados en los laboratorios.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

En el presente capítulo se realiza la descripción de la propuesta de solución de este trabajo, para ello se describen los procesos del negocio que tiene que ver con el objeto de estudio. Para entender mejor el contexto se realizará la modelación de los procesos del negocio con BPMN.

Además se enumeran los requisitos funcionales y no funcionales que debe tener el sistema que se propone, lo que permite hacer una concepción general del sistema e identificar mediante un Diagrama de Caso de Uso, las relaciones de los actores que interactúan con el sistema, y las secuencias de acciones que se realizan. También se elaborará una descripción en formato expandido de todos los casos de uso del sistema.

2.1. Objeto de automatización

Con el objetivo de facilitar el proceso de realización de los exámenes que se llevan a cabo en la UCI se hace necesario realizar un sistema informático que funcione como plataforma donde se puedan efectuar la mayoría de las acciones complementarias a dicho proceso. En este sistema serán objeto de automatización el proceso de realización de exámenes en los laboratorios, y el proceso de configuración de los mismos, los cuales a su vez traen implícitos la automatización de diversas acciones que antes se realizaban de forma manual o con el apoyo de otros medios informáticos, los cuales no se especializan en el control de dichas acciones.

2.2. Propuesta del Sistema

El sistema informático propuesto debe facilitar el proceso de realización de exámenes llevado a cabo en los laboratorios de la universidad. El desarrollo de la aplicación estará orientado a los exámenes que precisen ser evaluados total o parcialmente de forma digital, permitiendo realizar configuraciones sobre los mismos, además de realizar cálculos estadísticos, generar reportes, insertar calificaciones y otorgar permiso de descarga sobre el historial de exámenes. La interacción con el mismo será a través de una interfaz web, permitiendo ser accedido desde múltiples sistemas operativos.

2.3. Modelado del Negocio

Para describir la solución de software propuesta se realiza una representación del modelo del negocio, enfocado a la necesidad que existe en la universidad de efectuar el proceso de realización de exámenes de forma óptima y segura. Esta representación brinda una detallada explicación del funcionamiento de los procesos que componen dicho modelo, apoyándose en sus respectivos diagramas. El modelado utilizado fue BPMN, teniendo como proceso base la Realización de Exámenes en los Laboratorios.

2.3.1. Proceso base

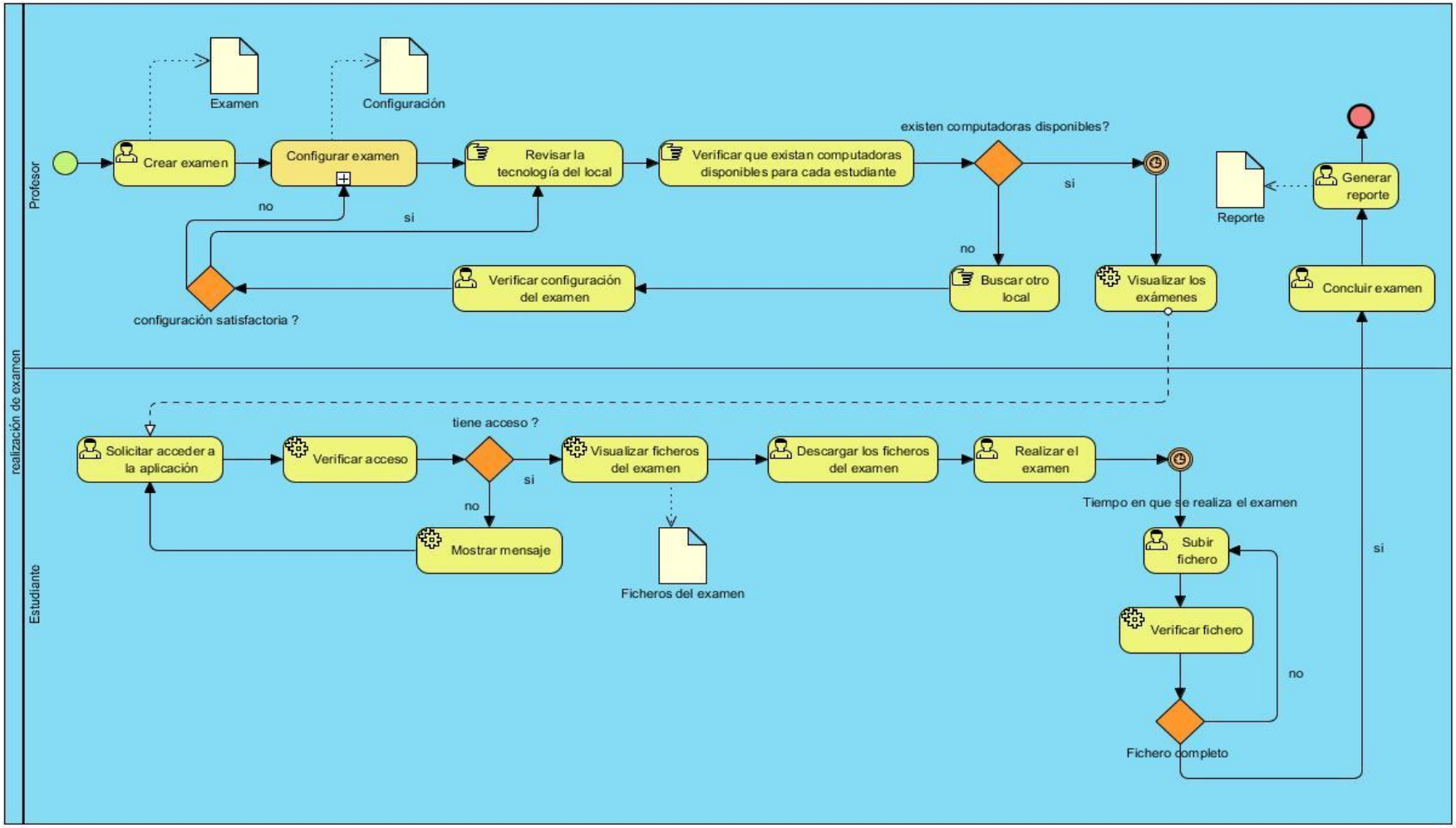


Figura 1: Realización de Examen en los Laboratorios

El proceso de realización de un examen tiene su inicio desde que se decide realizar un examen, ya sea a nivel central para todas las facultades o a nivel de facultad. El claustro de profesores de la asignatura del examen a aplicar es el responsable del proceso, tomando las decisiones necesarias para su correcta realización. Siempre se debe designar un profesor encargado de configurar el examen. En ambos casos existe una persona que es el encargado de crear el examen.

Tabla 1: Descripción del proceso base "Realización de examen en los laboratorios".

Nombre del proceso:	Realización de examen en los laboratorios.
Objetivo:	Evaluar los conocimientos impartidos en clases.
Precondiciones:	El usuario debe autenticarse en la aplicación.
Poscondiciones:	Debe quedar realizado el examen.
Responsables:	Profesor.
Entradas:	No presenta.
Salidas:	Reporte.
Subprocesos:	Configurar examen.
Actividades	
Crear examen: El profesor responsable de aplicar el examen al grupo debe crear previamente el examen en el sistema**.**	
Configurar examen: El profesor responsable de aplicar el examen debe configurarlo previamente.	
Revisar la tecnología del local: El profesor responsable de aplicar el examen al grupo, debe revisar previamente la tecnología del local.	
Verificar que existen computadoras disponibles para cada estudiante: Tras haber revisado la tecnología, el profesor responsable realiza un conteo manual de cada computadora por estudiante.	

Buscar otro local: En caso de que no alcance la tecnología en el local, el profesor responsable se verá en la obligación de buscar otro local disponible para los alumnos que no alcanzaron computadoras.
Verificar configuración del examen: En caso de haber trasladado alumnos a otro local, el profesor verifica si la configuración del examen es válida.
Visualizar examen: El sistema verifica el tiempo de inicio del examen y si se corresponde con el actual procede a visualizarlo.
Establecer fecha de visibilidad: Tras haber configurado las opciones de acceso se procede a fijar la visibilidad del fichero, que puede ser manual o automática**.**
Solicitar acceder a la aplicación: El estudiante ingresa sus datos en el sistema y presiona el botón acceder.
Verificar acceso: El sistema verifica los datos del estudiante una vez ingresados.
Mostrar mensaje: El sistema muestra un mensaje de alerta en caso de que los datos insertados sean incorrectos.
Visualizar ficheros del examen: El sistema visualiza los ficheros del examen una vez llegado el momento de comienzo del mismo.
Descargar los ficheros del examen: El estudiante descarga los ficheros del examen tras estar visualizados.
Realizar el examen: El estudiante realiza el examen.
Subir fichero: El estudiante sube el fichero resultante a la aplicación una vez terminado el examen.
Verificar fichero: El sistema verifica que el fichero subido se haya modificado o no se encuentre vacío.
Concluir examen: El profesor concluye el examen una vez que todos los alumnos hayan subido sus archivos.
Generar reporte: Tras haber concluido el tiempo de realización del examen el profesor encargado del grupo genera un reporte del mismo**.**

2.3.2. Subproceso "Configurar Examen"

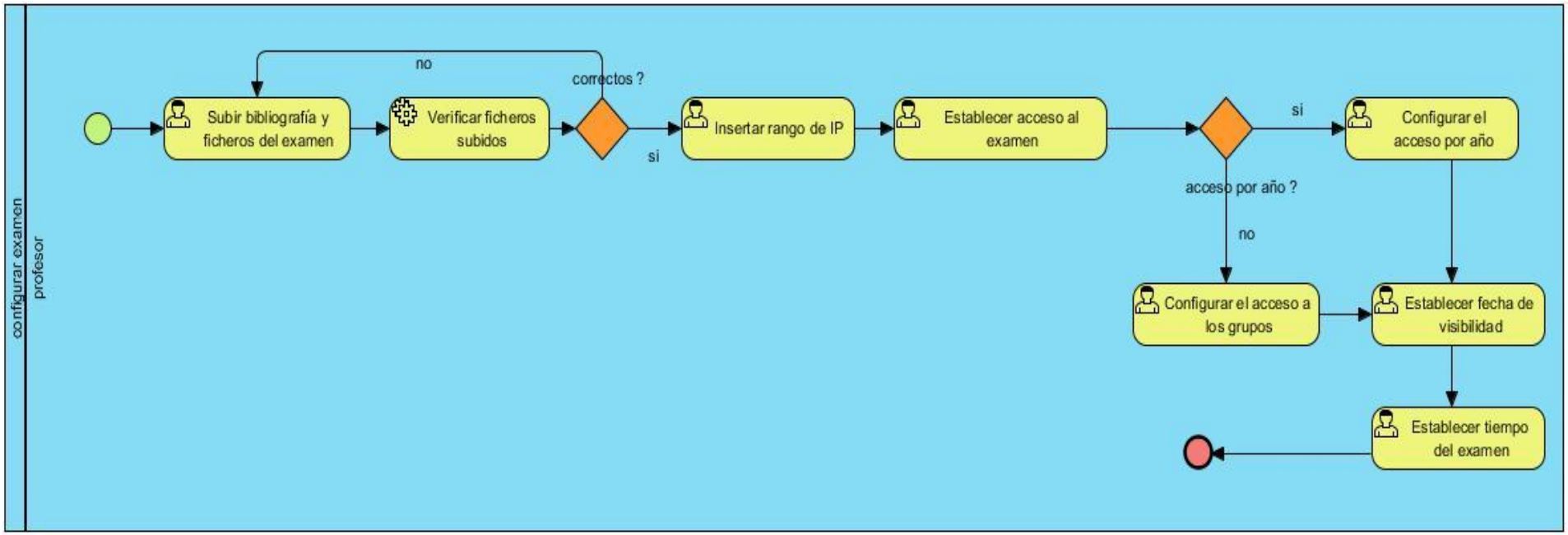


Figura 2: Diagrama del negocio del subproceso "Configurar Examen".

Tabla 2: Descripción del subproceso "Configurar examen".

Nombre del proceso:	Configurar Examen
Objetivo:	Facilitar la realización de los exámenes en los laboratorios, así como aumentar y mantener la seguridad sobre los mismos, tratando de evitar el fraude académico.
Precondiciones:	El usuario debe haberse autenticado en la aplicación como profesor
Poscondiciones:	Debe quedar aplicada la configuración.
Responsables:	Profesor
Entradas:	Rango de IP válidos, fecha de inicio del examen, hora de inicio del examen, tiempo de duración, niveles de organización y visualización manual o automática.
Salidas:	Configuración de examen
Subprocesos	No presenta
Actividades	
Subir la bibliografía y fichero del examen: Se adjunta el fichero del examen y la bibliografía complementaria al mismo, proveyendo todos los materiales necesarios para la realización del mismo.	
Verifica los ficheros subidos: Tras haber subido el examen y la bibliografía complementaria al examen, el sistema verifica que los ficheros se hallan subido correctamente, y depende al resultado emite una alerta.	
Insertar rango de direcciones IP: Una vez que los ficheros estén subidos correctamente a la aplicación el usuario puede insertar el rango de direcciones IP validos desde los cuales se pueden acceder al examen en la aplicación.	
Establecer acceso a la prueba: Se establece el acceso al examen de acuerdo el año, facultad o grupo.	

Configurar acceso por año: Se establece el año que realizará el examen.
Configurar el acceso por grupo: Se establecen los grupos que realizarán el examen.
Establecer fecha de visibilidad: Tras haber configurado las opciones de acceso se procede a fijar la visibilidad del fichero, que puede ser manual o automática.
Establecer el tiempo del examen: Se establece el tiempo indicado para la realización del examen.

2.4. Reglas del negocio

1. **Identificación:** Todo usuario debe autenticarse en la aplicación para acceder a realizar las acciones pertinentes, así como para obtener los ficheros o datos requeridos.
2. **Responsabilidad:** El profesor encargado de realizar el examen podrá configurar y calificar un examen solo a la asignatura, los años y grupos que atiende.

   El profesor debe ser el encargado de verificar con un tiempo mínimo de 1 día a realizar un examen; que la tecnología se encuentre lista y en correspondencia con la cantidad de estudiantes.
3. **Descargar archivos del historial:** Para descargar algún archivo del historial, el alumno deberá requerir los permisos necesarios para realizar esta acción.
4. **Configuración:** Para configurar un examen debe estar previamente creado en el sistema.

2.5. Especificación de Requisitos

La IEEE (*Institute of Electrical and Electronic Engineers,* (en español Instituto de Ingenieros Eléctricos y Electrónicos)) define al requisito como la condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente (28). Los requisitos son una parte importante en el desarrollo de aplicaciones informáticas, debido a que una mala definición, especificación o administración de los mismos puede conllevar al fracaso del sistema.

En esta sección se define lo que debe hacer el sistema y las características con las que debe cumplir. Para esto se identifican las funcionalidades requeridas y las restricciones que se imponen, teniendo en cuenta la clasificación en requisitos funcionales y no funcionales.

2.5.1. Requerimientos funcionales

Los requisitos funcionales indican el comportamiento del sistema.

RF-1. Autenticar usuario:

El sistema debe identificar los usuarios, y permitir el acceso de los mismos una vez identificados.

- Usuario
- Contraseña

RF-2. Solicitar acceso al sistema:

El sistema debe brindar la posibilidad a los profesores de realizar una solicitud de acceso al sistema.

- Usuario
- Solapín
- Años a los que imparte clases
- Grupos que atiende
- Facultad
- Correo
- Nombre

RF-3. Insertar examen:

El sistema debe brindar la posibilidad de crear un nuevo examen.

- Nombre del examen
- Asignatura del examen

RF-4. Modificar examen:

El sistema debe ser capaz de permitir a un usuario con los permisos necesarios cambiar el nombre del examen una vez creado.

- Nombre del examen

RF-5. Eliminar examen:

El sistema debe brindar la opción de eliminar un examen ya creado.

RF-6. Configurar examen:

El sistema debe permitir configurar un examen con los datos específicos mostrados en el sistema.

- Rango de direcciones IP
- Fecha de inicio de examen
- Hora de inicio del examen
- Tiempo de duración
- Niveles de organización
- Visualización del examen

RF-7. Subir fichero:

El sistema debe ofrecer la opción de subir el fichero del examen a la aplicación.

RF-8. Aplicar configuración:

El sistema debe permitir aplicar la configuración determinada por el usuario.

RF-9. Configurar visibilidad:

El sistema debe permitir mostrar u ocultar un fichero terminado un examen.

RF-10. Insertar calificación:

El sistema debe permitir insertar una calificación a un determinado examen.

RF-11. Buscar examen:

El sistema debe permitir realizar una búsqueda y un listado de los archivos de exámenes dado el nombre del mismo.

- Nombre del examen.

RF-12. Establecer permiso de descarga del historial de exámenes:

El sistema debe permitir a un profesor establecer un permiso de descarga sobre el historial de exámenes de un estudiante determinado.

RF-13. Enviar notificación:

El sistema debe permitir emitir un aviso a determinado usuario a través de una notificación al correo electrónico.

RF-14. Generar reporte:

El sistema debe permitir generar un reporte con el desempeño de la realización del examen.

RF-15. Registrar trazas:

El sistema debe ser capaz de crear un fichero con todas las trazas dejadas por los estudiantes en el sitio.

RF-16. Realizar cálculos estadísticos:

El sistema debe ser capaz de realizar datos estadísticos en relación al tiempo de realización del examen.

RF-17. Enviar solicitud de descarga del historial:

El sistema debe permitir realizar el envío de una solicitud de descarga del historial de exámenes de un determinado estudiante.

RF-18. Descargar examen del historial:

El sistema debe permitir realizar un listado de todos los exámenes hechos anteriormente por los estudiantes.

RF-19. Adicionar asignatura:

El sistema debe permitir adicionar una nueva asignatura en la base de datos.

- Nombre de la asignatura
- Año

RF-20. Modificar asignatura:

El sistema debe permitir modificar los datos de una asignatura existente en la base de datos.

- Nombre de la asignatura
- Año

RF-21. Buscar asignatura:

El sistema debe permitir realizar una búsqueda de las asignaturas existentes en la base de datos.

- Año

RF-22. Eliminar asignatura:

El sistema debe permitir eliminar una asignatura de la base de datos.

- Nombre de la asignatura

RF-23. Adicionar un nuevo usuario:

El sistema debe permitir añadir un nuevo usuario, el cual ocupará determinado rol.

- Usuario
- Correo
- Rol
- Facultad

RF-24. Modificar un usuario:

El sistema debe permitir cambiar el rol de un usuario.

- Usuario
- Correo
- Rol
- Facultad

2.6. Requisitos no funcionales

Se puede entender por requisitos no funcionales a las propiedades o cualidades que el sistema debe poseer. Estas características son las que hacen del sistema un producto atractivo, usable, rápido o confiable, conllevando a su éxito.

Las categorías en las que se pueden clasificar los requisitos no funcionales son:

2.6.1. Requisitos de Hardware:

El sistema desarrollado es una aplicación web, por lo que empleará recursos hardware tanto en el lado del cliente como del servidor.

En el lado del cliente el consumo de los recursos de hardware recae sobre el microprocesador, la memoria RAM y las conexiones de red. Debido a que solo se necesita un navegador web para acceder a la aplicación, es necesario poseer los recursos hardware mínimo e indispensable para que el mismo se ejecute.

A continuación se establecen los recursos hardware que necesitará el cliente:

- Un microprocesador Pentium 4 a 2.4 GHz.

- Una capacidad de almacenamiento mínimo de memoria RAM de 512 Mb (Megabytes).
- Las conexiones de red pueden ser establecidas mediante un cable UTP con conectores RJ45 y una tarjeta de red Ethernet a 100 Mbps o superior. Además puede ser utilizado el acceso inalámbrico.

Del lado del servidor quedan establecidos los siguientes recursos hardware:

- Un microprocesador Pentium 4 a 3.0 GHz.
- Una capacidad de almacenamiento mínimo de memoria RAM de 1 Gb (Gigabytes).
- Las conexiones de red pueden ser establecidas mediante un cable UTP con conectores RJ45 y una tarjeta de red Ethernet a 100 Mbps o superior. Además puede ser utilizado el acceso inalámbrico.
- El sistema se diseñó para almacenar archivos de exámenes y otros recursos esenciales. Por lo que se define como requisito mínimo de capacidad de almacenamiento 5GB de espacio libre en el disco duro en el que se encuentre el sistema.

2.6.2. Requisitos de Software:

Debido a que la aplicación se desarrolla con el Framework Symfony2, el cual se utiliza del lado del servidor, y el mismo presenta ciertos requisitos para su instalación, los requerimientos de software son los especificados.

- El sistema se desarrollará con tecnología PHP versión 5.4.3
- El sistema se desplegará en servidores GNU/Linux, con sistemas operativos Ubuntu 11.10 o superior. Además puede ser publicado en servidores con sistemas operativos Windows 7 u 8.
- Se utilizará tecnología Apache 2.2.22 para el servidor web
- El sistema utilizará el Gestor de Base de Datos MySQL 5.5.24.
- Al poseer una interfaz Web las computadoras clientes pueden tener instalada cualquier versión del sistema operativo GNU/Linux, o sistema operativo Windows 7 u 8.
- Debido a que la aplicación se desarrolla con tecnología avanzada, se requiere un navegador Mozilla Firefox que tenga compatibilidad con HTML5 y CSS3, por

lo que se recomienda una versión actualizada del mismo. También se pueden utilizar Internet Explorer 10, Chrome en su versión 28, Opera *12* y Safari 5.1.

2.6.3. Requisitos de apariencia o interfaz externa:

Una vez accedido el usuario a la aplicación, las opciones correspondientes al mismo se mostrarán en un menú principal. Dichas opciones son obtenidas de acuerdo al rol ocupado por el usuario autenticado, y el diseño estará relacionado por plantillas con un mismo estilo.

2.6.4. Requisitos de Seguridad:

La aplicación contará con la seguridad ofrecida por el *Framework* Symfony2, el cual configura todos los parámetros de dicho requisito en el archivo “app/config/security.yml”. La seguridad se basa en roles, asignados a cada usuario de acuerdo a su categoría, la cual puede ser “Administrador”, “Profesor”, o “Estudiante”.

2.6.5. Requisitos de Usabilidad:

Mensajes del sistema en la interfaz: Todos los mensajes que interactúen con el usuario, mostrando una determinada información, deberán ser lo suficientemente claros y explicativos. Además el lenguaje de los mensajes deberá ser en idioma castellano.

Navegación de los usuarios en el sistema: cada interfaz muestra la información relacionada con el usuario autenticado en el sistema, debido a esto la información que se maneja es conocimiento del usuario autenticado. Además no es necesario un entrenamiento en la navegación del sistema, ya que cada interfaz posee un mensaje de información que explica su funcionamiento.

Interfaces de usuario: Todas las interfaces que se muestren, al igual que los botones asociados a las mismas deben poseer un color común, caracterizándose e identificándose como interfaces del sistema.

2.6.6. Portabilidad, escalabilidad:

En la universidad existe una gran variedad de computadoras con diferentes sistemas operativos, por lo que la aplicación debe ser multiplataforma y de esta manera cumplir con el objetivo propuesto. Además el sistema está diseñado para ser portable y configurable, definiendo su usabilidad desde un nivel central para su uso en toda la universidad, hasta un nivel de facultad, en la cual se le puede aplicar el examen a un grupo específico.

2.6.7. Confiabilidad y fiabilidad:

La aplicación debe garantizar la confiabilidad y fiabilidad de los datos, almacenando en el servidor toda la información recolectada, además se efectuarán resguardos de la misma y se harán salvas de la base de datos periódicamente (*Backups*), garantizando así la estabilidad y movilidad del sistema en caso de algún fallo en el sistema operativo o en el hardware. Toda la información es almacenada en el servidor, en un sitio predeterminado por el sistema, siguiendo una estructura establecida, por lo que debe ser fácil de acceder o manejar en caso de urgencia u ocurrencia de problemas de diversas índoles. La información es trasmitida de los clientes al servidor mediante el protocolo HTTP, usando específicamente el método de envío post, el cual envía las variables a través de un formulario, sin ser visibles en la URL (sigla en inglés de *Uniform Resource Locator*) del navegador.

2.7. Modelo de Casos de Uso del Sistema

Tras haber identificado y clasificado los requerimientos del sistema se hace necesario identificar los actores y casos de uso correspondiente al mismo.

2.7.1. Definición de los actores del sistema a automatizar

Tabla 3: Actores del Sistema

Actor del Sistema	Descripción
Profesor	Es el encargado de realizar los exámenes correspondientes a la asignatura que atiende, así como de configurarlos y calificarlos. Además de otorgar permisos a las solicitudes hechas por los estudiantes para descargar un determinado examen del historial.
Administrador del Sistema	Es el encargado de hacer las gestiones en la aplicación, ya sea de insertar en la base de datos a los profesores vinculados a las asignaturas como de gestionar las asignaturas.
Estudiante	Descarga y sube los ficheros del examen a la aplicación una vez terminada la evaluación y puede enviar una solicitud de descarga en caso que necesite descargar un examen del historial.

2.7.2. Diagrama de Casos de Uso:

Los casos de uso son los objetivos que un usuario quiere lograr con un sistema, estos describen y documentan el comportamiento del sistema desde el punto de vista de un usuario (29).

A partir del proceso de modelado del negocio y la especificación de requisitos funcionales del sistema se identifican los casos de usos que representan el flujo de eventos definidos por los actores.

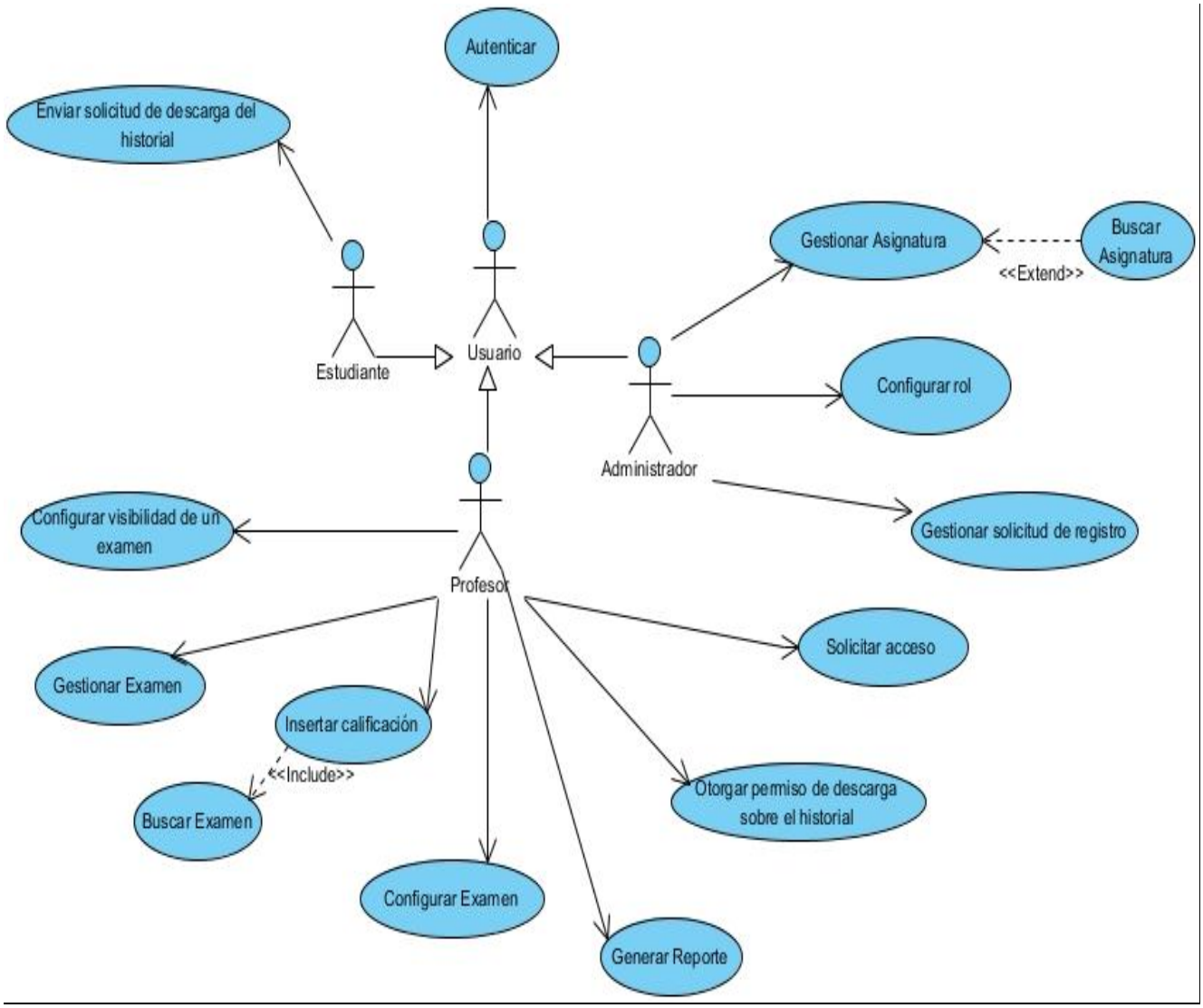


Figura 3: Diagrama de Casos de Uso del Sistema.

2.8. Descripción textual de los Casos de Uso del Sistema

A continuación se exponen las descripciones de los casos de uso del sistema, para lograr un mejor entendimiento de los procesos globales de la aplicación.

CU-1: Autenticar

Tabla 4: Caso de uso Autenticar

Caso de Uso	Autenticar
Actores	Usuario
Resumen	El usuario se autentica y posteriormente se le otorga el acceso y permiso correspondiente al rol que ocupa.
Complejidad	Baja
Prioridad	Crítico
Precondiciones	El usuario tiene que ser un profesor, administrador o estudiante.
Referencias	RF1
Flujo normal de eventos	
Sección "Autenticar usuario".	
Acción del actor	**Respuesta del sistema**
1- El usuario ingresa los datos solicitados. • Usuario • Contraseña	
2- El usuario presiona el botón "Entrar"	3- El sistema se conecta al dominio UCI haciendo uso de la librería LDAP y verifica que los datos ingresados sean correctos.
	4- El sistema registra los datos del usuario en la base de datos.
	5- El sistema muestra la página principal correspondiente a su rol.

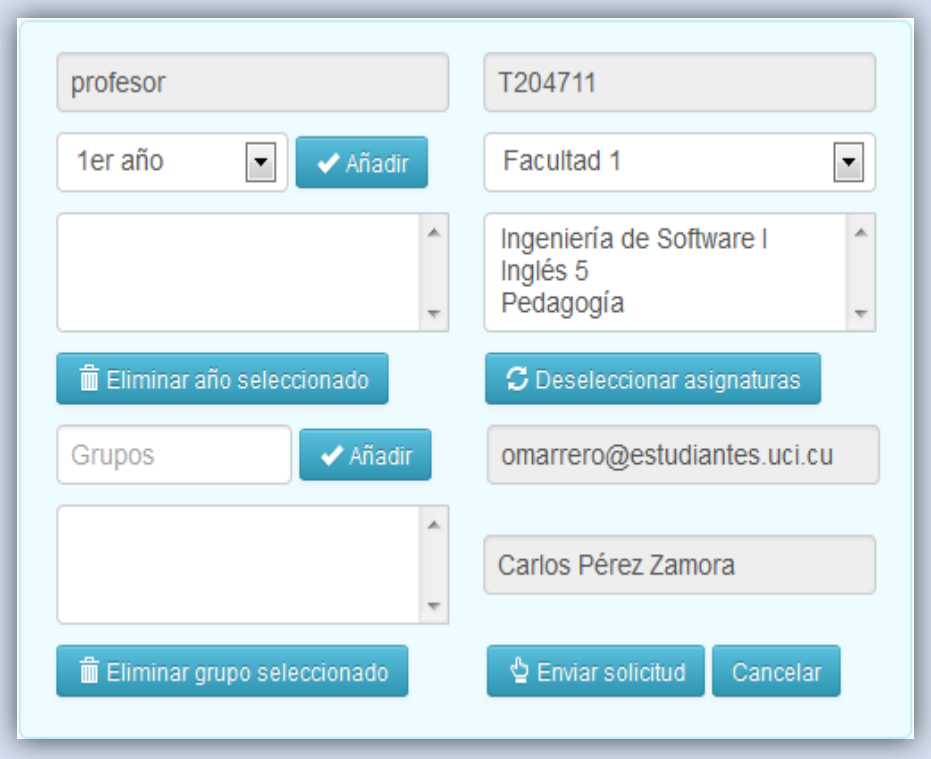
Prototipo de Interfaz	
	
Flujo alterno 1	
Acción del actor	**Respuesta del sistema**
2.1- El usuario presiona el botón “Entrar”.	3.1- El sistema se conecta al dominio UCI con la librería LDAP y verifica que los datos sean correctos.
	4.1- El sistema muestra el mensaje “Datos Incorrectos!!!”.
Prototipo de Interfaz	
	
Poscondiciones	

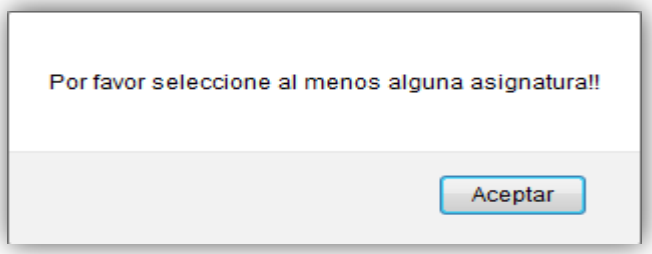
CU-2: Solicitar Acceso.

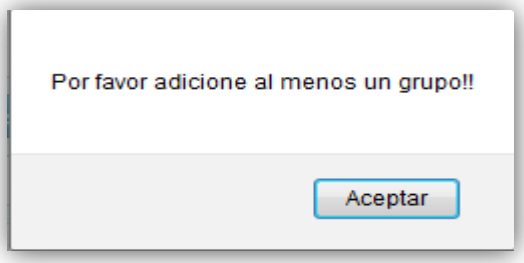
Tabla 5: Caso de Uso Solicitar Acceso.

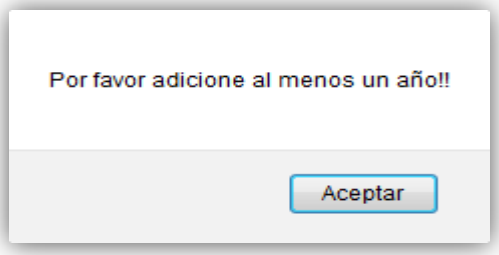
Caso de Uso	Solicitar Acceso.
Actores	Profesor
Resumen	El profesor solicita el acceso a la aplicación, ingresando los datos necesarios para su desempeño con el sistema.
Complejidad	Baja
Prioridad	Crítico.
Precondiciones	1- El usuario debe estar autenticado en el sistema como profesor. 2- La solicitud del profesor no debe estar aceptada.
Referencias	RF2
Flujo normal de eventos	
Sección "Solicitud de acceso".	
Acción del actor	**Respuesta del sistema**
1- El profesor ingresa los datos necesarios para conformar una solicitud de acceso. • Usuario • Solapín • Años • Grupos • Facultad • Correo	

• Nombre • Asignaturas	
2- El profesor presiona el botón “Enviar solicitud”.	3- El sistema verifica que no hayan campos vacíos.
	4- El sistema registra una nueva solicitud de acceso en la base de datos.
	5- El sistema oculta los campos de registro y muestra el mensaje “Su solicitud ha sido enviada…esperando respuesta del administrador”.
Prototipo de Interfaz	



<table>
<tr><th colspan="2">Flujo alterno 1</th></tr>
<tr><th>Acción del actor</th><th>Respuesta del sistema</th></tr>
<tr><td></td><td>4.1- El sistema muestra un mensaje de error en dependencia de los campos vacíos:
“por favor seleccione al menos alguna asignatura”.
“por favor adicione al menos un grupo!!”.
“por favor adicione al menos un año!!”.</td></tr>
<tr><th colspan="2">Prototipo de Interfaz</th></tr>
<tr><td colspan="2"> 

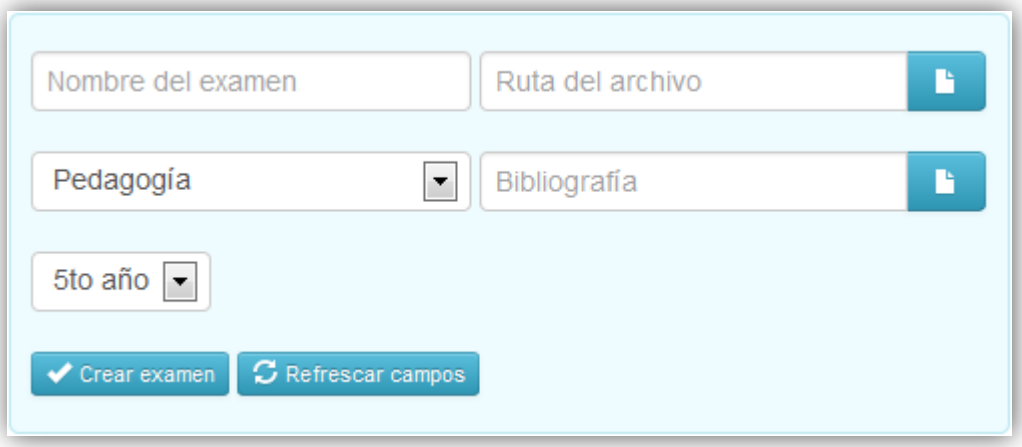



</td></tr>
<tr><th>Poscondiciones</th><td></td></tr>
</table>

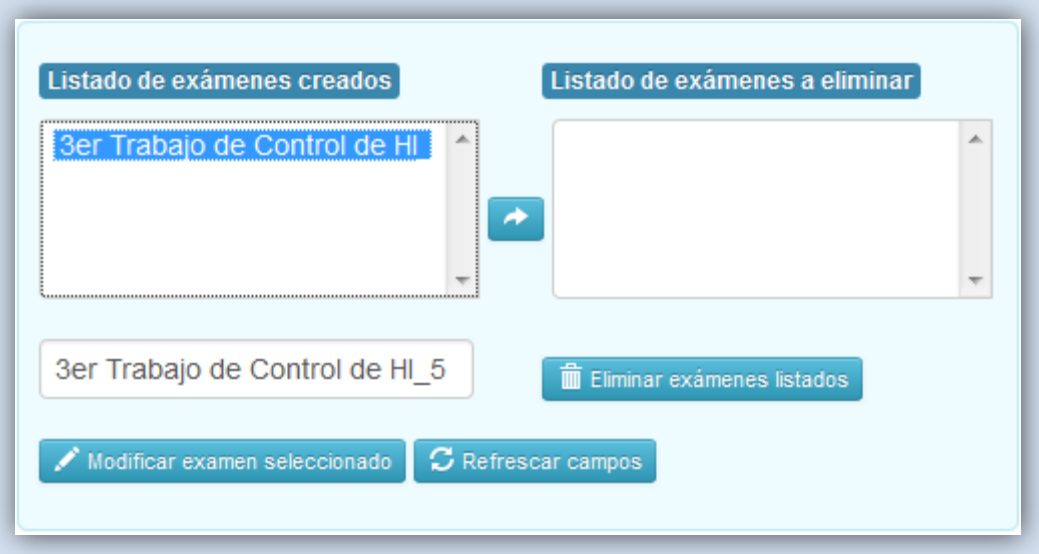
CU-3: Gestionar Examen

Tabla 6: Caso de Uso Gestionar Examen

Caso de Uso	Gestionar Examen
Actores	Profesor
Resumen	Permite adicionar, modificar y eliminar un examen.
Precondiciones	El profesor debe estar autenticado en el sistema.
Referencias	RF3, RF4, RF5
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1- El profesor selecciona la opción “Gestionar examen”.	2- El sistema redirecciona al profesor a la página Gestionar examen.
Sección “Insertar examen”.	
3- El profesor inserta los datos del examen: • Nombre del examen • Asignatura al que pertenece el examen. • Posteriormente adiciona el archivo del examen y la bibliografía anexa.	
4- El profesor presiona el botón “Crear examen”.	5- El sistema verifica que los datos hayan sido entrados correctamente.
	6- El sistema almacena los datos en la base de datos.
	7- El sistema redirecciona al profesor a la página “Principal”.

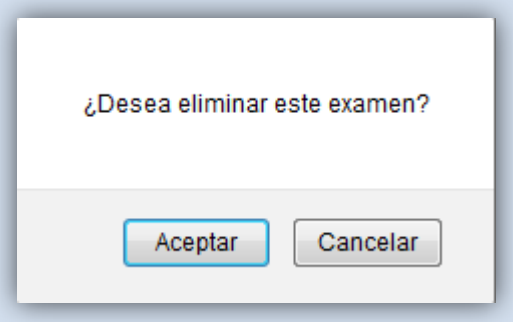
Prototipo de Interfaz


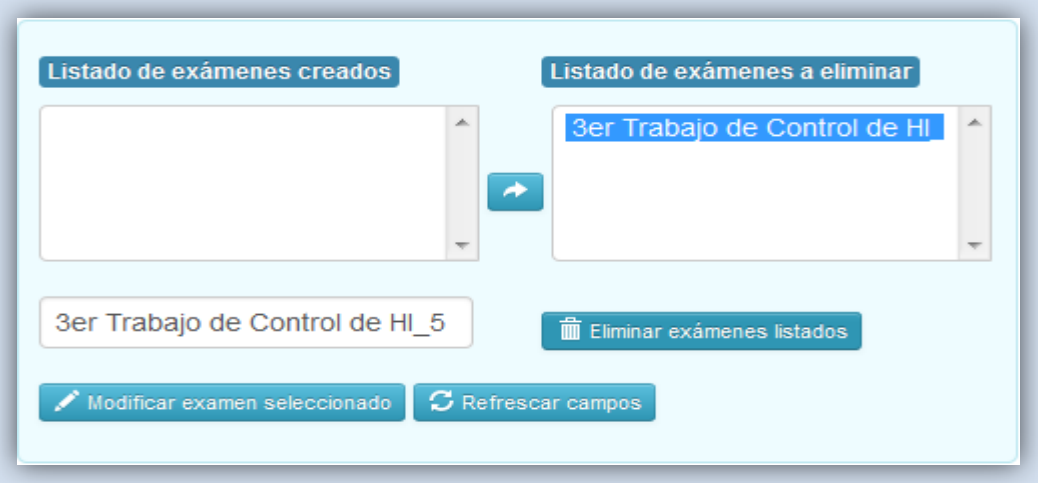
Flujo alterno 1	
Acción del actor	**Respuesta del sistema**
4.1- El profesor presiona el botón “Refrescar campos”.	5.1- El sistema pone en blanco todos los campos del formulario Insertar examen.
Prototipo de Interfaz	
Sección “Modificar examen”.	
1- El profesor selecciona el examen a modificar.	2- El sistema obtiene el nombre del examen seleccionado y lo inserta en el campo de texto nuevo nombre
3- El profesor inserta un nuevo nombre correspondiente al examen seleccionado.	
4- El profesor presiona el botón “Modificar examen seleccionado”.	5- El sistema busca en la base de datos el examen seleccionado y le modifica el nombre.
Prototipo de Interfaz	

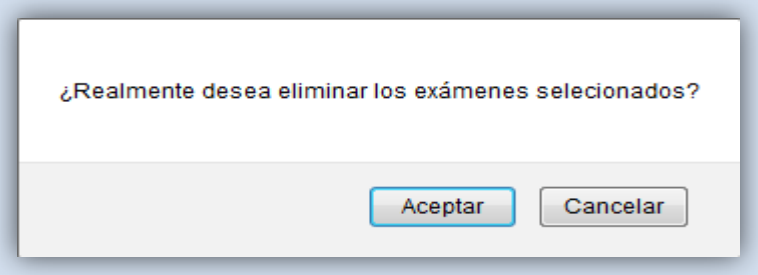


Flujo alterno 1	
Acción del actor	**Respuesta del sistema**
4.1- El profesor presiona el botón “Refrescar campos”.	5.1- El sistema pone en blanco el campo de texto nuevo nombre y vuelve a listar los exámenes realizados.
Prototipo de Interfaz	
Sección “Eliminar examen”.	
1- El profesor selecciona un examen de la lista de exámenes realizados.	
2- El profesor selecciona un examen y presiona el botón cambiar a lista de eliminados.	3- El sistema muestra un mensaje de confirmación con el texto “Desea eliminar este examen…”.
4- El profesor confirma el mensaje del sistema.	5- El sistema mueve el examen seleccionado a la lista de exámenes a eliminar.
6- El profesor presiona el botón “eliminar exámenes	7- El sistema muestra el cartel “Realmente desea

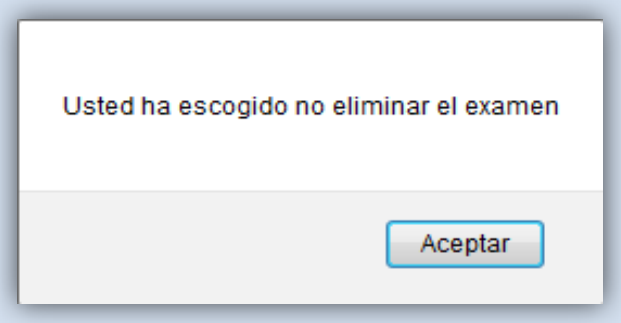
listados”.	eliminar los exámenes seleccionados”.
Prototipo de Interfaz	







Flujo alterno 1	
4.1- El profesor no confirma el mensaje del sistema.	5.1- El sistema muestra un mensaje con el texto “Usted ha escogido no eliminar el examen”.
Prototipo de Interfaz	



Poscondiciones	

Conclusiones

Como parte del presente capítulo se abordó todo lo referente al funcionamiento y estructura del negocio del sistema, y se utilizó como modelado del mismo, BPMN. También se definieron las reglas del negocio y se especificaron los requerimientos con los que tenía que cumplir la aplicación. Además se mostraron los diagramas de casos de uso del sistema y las descripciones de los mismos.

CAPÍTULO 3: DISEÑO DEL SISTEMA

Este capítulo se centra en describir la solución del sistema a través del diseño de la aplicación, obteniendo como resultado los artefactos más importantes que describen cómo implementar el sistema. Se realizan los diagramas de clases de diseño y de interacción. También se describen los estilos arquitectónicos, así como los patrones de diseño empleados y el modelo de datos utilizado.

3.1. Estilo arquitectónico

En el texto fundacional de la Arquitectura de Software, Perry y Wolf establecen el razonamiento sobre estilos de arquitectura como uno de los aspectos fundamentales de la disciplina. Un estilo es un concepto descriptivo que define una forma de articulación u organización arquitectónica. El conjunto de los estilos cataloga las formas básicas posibles de estructuras de software, mientras que las formas complejas se articulan mediante composición de los estilos fundamentales (30).

Cada estilo arquitectónico describe una categoría del sistema que contiene, ya sea un conjunto de componentes, que realiza una función requerida por el sistema, un conjunto de conectores que posibilitan la comunicación, la coordinación y la cooperación entre los componentes; restricciones que definen como se puede integrar los componentes que forman el sistema; y modelos semánticos que permiten al diseñador entender las propiedades globales de un sistema para analizar las propiedades conocidas de sus partes constituyentes.

Los estilos arquitectónicos definen los patrones posibles de las aplicaciones. Permiten evaluar arquitecturas alternativas con ventajas y desventajas conocidas ante diferentes conjuntos de requerimientos no funcionales, así como sintetizar estructuras de soluciones (31).

¿Qué es un Patrón de Arquitectura?

Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de software, que consta de subsistemas, sus responsabilidades e interrelaciones (32).

3.1.1. Arquitectura Cliente/Servidor:

La Arquitectura Cliente/Servidor es un modelo para el desarrollo de sistemas de información en el que las transacciones se dividen en procesos independientes que cooperan entre sí para intercambiar información, servicios o recursos.

Cliente: Es el que inicia el diálogo o solicita los recursos.

Servidor: Es el proceso que responde a las solicitudes.

En el modelo Cliente/Servidor las aplicaciones se dividen de forma que el servidor contiene la parte que debe ser compartida por varios usuarios, y en el cliente permanece sólo lo particular de cada usuario.

Características de la arquitectura Cliente/Servidor:

- El proceso del cliente proporciona la interfaz entre el usuario y el resto del sistema. El proceso del servidor actúa como un motor de software que maneja recursos compartidos tales como bases de datos, impresoras, entre otros.
- Existen diferencias de requerimientos en cuanto a recursos de cómputo que consumen las tareas en el cliente y el servidor, como por ejemplo velocidad del procesador, memoria, velocidad y capacidades del disco, entre otras.
- Se establece una relación entre procesos distintos, los cuales pueden ser ejecutados en una misma máquina o en máquinas diferentes distribuidas a lo largo de la red.
- La relación establecida puede ser de muchos a uno, en la que un servidor puede dar servicio a muchos clientes, regulando su acceso a recursos compartidos.
- Los clientes corresponden a procesos activos ya que son éstos los que hacen peticiones de servicios a los servidores. Los servidores tienen un carácter pasivo debido que esperan las peticiones de los clientes.
- No existe otra relación entre clientes y servidores que no sea la que se establece a través del intercambio de mensajes entre ambos. El mensaje es el mecanismo para la petición y entrega de solicitudes de servicio.
- El ambiente es heterogéneo. La plataforma de hardware y el sistema operativo del cliente y del servidor no son siempre la misma (33).

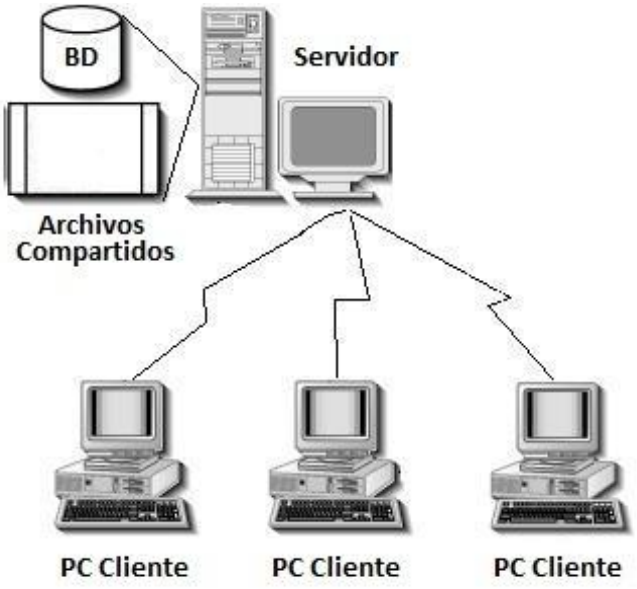


Figura 4. Arquitectura Cliente/Servidor.

Dada la existencia de múltiples estilos y patrones arquitectónicos, en la realización del sistema a desarrollar se decidió utilizar el patrón arquitectónico **Modelo-Vista-Controlador** (MVC).

3.1.2. Patrón arquitectónico utilizado (Modelo Vista Controlador)

El patrón **Modelo-Vista-Controlador** separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Este patrón es utilizado por Symfony2 en su funcionamiento, y se implementa como se muestra a continuación.

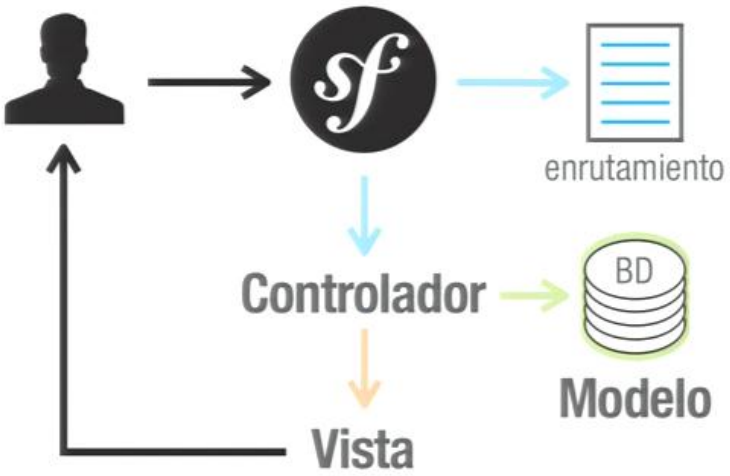


Figura 5. Patrón arquitectónico MVC en Symfony2.

Un modelo puede tener diversas vistas, cada una con su correspondiente controlador, aunque es posible definir múltiples vistas para un único controlador (34).

La arquitectura MVC en Symfony2 separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. Si por ejemplo una misma aplicación debe ejecutarse tanto en un navegador estándar como en un navegador de un dispositivo móvil, solamente es necesario crear una vista nueva para cada dispositivo; manteniendo el controlador y el modelo original (35).

El principio más importante de la arquitectura MVC es la separación del código del programa en tres componentes, dependiendo de su naturaleza. La lógica relacionada con los datos se incluye en el modelo, el código de la presentación en la vista y la lógica de la aplicación en el controlador.

El modelo: Es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El modelo no tiene conocimiento específico de los controladores o de las vistas, debido que no posee referencias a ellos. Es el propio sistema el que posee la responsabilidad de mantener enlaces entre el modelo y sus vistas, y notificar a las vistas cuando cambia el modelo. En Symfony2 se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación.

La vista: Es el objeto que maneja la presentación visual de los datos representados por el modelo. Genera una representación visual del modelo y muestra los datos al usuario. Interactúa con el controlador. En Symfony2 la capa de la vista también puede aprovechar la separación de código. Las páginas web suelen contener elementos que se muestran de forma idéntica a lo largo de toda la aplicación: cabeceras de la página, el *layout* genérico, el pie de página y la navegación global. Normalmente sólo cambia el interior de la página. Por este motivo, la vista se separa en un *layout* y en una plantilla. Normalmente, el *layout* es global en toda la aplicación o al menos en un grupo de páginas. La plantilla sólo se encarga de visualizar las variables definidas en el controlador. Para que estos componentes interaccionen entre sí correctamente, es necesario añadir cierto código.

El controlador: Es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el modelo, centra toda la interacción entre la vista y el modelo. Cuando se realiza algún cambio, entra en acción, sea por cambios en la información del modelo o por alteraciones de la vista. Interactúa con el modelo a través de una referencia al propio modelo. En Symfony2 se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, entre otros.).

3.1.3. Diseño de la arquitectura del sistema.

En el sistema para la configuración de exámenes que se realizan en los laboratorios, las reglas esenciales se administrarán del lado del servidor, apartando en el lado del cliente las reglas de presentación y validación de la información. De esta forma el diseño propone que los usuarios accedan a través de un navegador web a las funcionalidades de un sistema centralizado.

El diseño que se propone, encapsula y delimita las responsabilidades de cada una de las partes del sistema en varios componentes esenciales. La figura a continuación representa el diseño de la arquitectura del sistema.

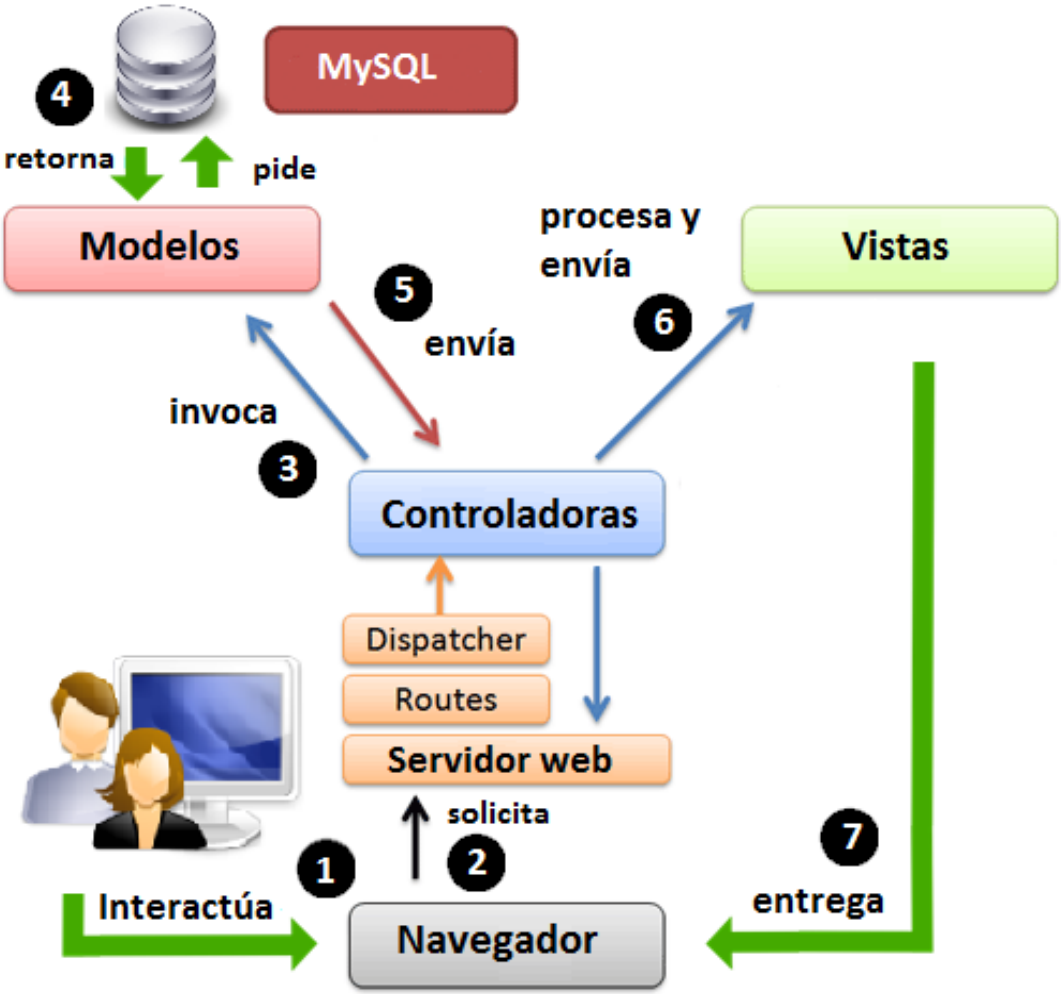


Figura 6. Arquitectura del sistema (MVC).

En este modelo el *framework* Symfony2 gestiona los datos haciendo uso del ORM (*Object Relational Mapper* de sus siglas en inglés) Doctrine2, el cual traduce las clases (Entidades del negocio) a tablas y las tablas a clases. De esta forma se manejan las entidades como si fueran tablas en la base de datos. En el *framework* la capa del modelo se puede dividir en la capa de acceso a los datos y en la capa de abstracción de la base de datos. De esta forma, las funciones que acceden a los datos no utilizan

sentencias ni consultas que dependen de una base de datos, sino que utilizan otras funciones para realizar las consultas. Así, si se cambia de sistema gestor de bases de datos, solamente es necesario actualizar la capa de abstracción de la base de datos.

Entidades del negocio: Permiten la abstracción de la realidad. Son las entidades que modelan el problema.

Seguridad en Symfony2

En el sistema de seguridad se presentan las funcionalidades que garantizan el nivel de acceso de los usuarios al sistema, verificando en cada petición que el usuario tenga permiso para ejecutarla. La seguridad en Symfony2 es un proceso de dos etapas, cuyo objetivo es evitar que un usuario acceda a un recurso al cual no debería tener acceso (36).

En la primera etapa del proceso, el sistema de seguridad identifica quién es el usuario obligándolo a presentar algún tipo de identificación. Esto se llama autenticación, y significa que el sistema está tratando de determinar quién es, esto es llevado a cabo por el firewall del sistema.

Una vez que el sistema haya identificado al usuario, la siguiente etapa es determinar si tiene acceso a un determinado recurso. Esta parte del proceso se llama autorización, y significa que el sistema está comprobando si tienes suficientes privilegios para realizar una determinada acción. Esta acción se encuentra determinada por las listas de control de acceso establecidas en el fichero de seguridad. La figura a continuación representa el sistema de seguridad del sistema.

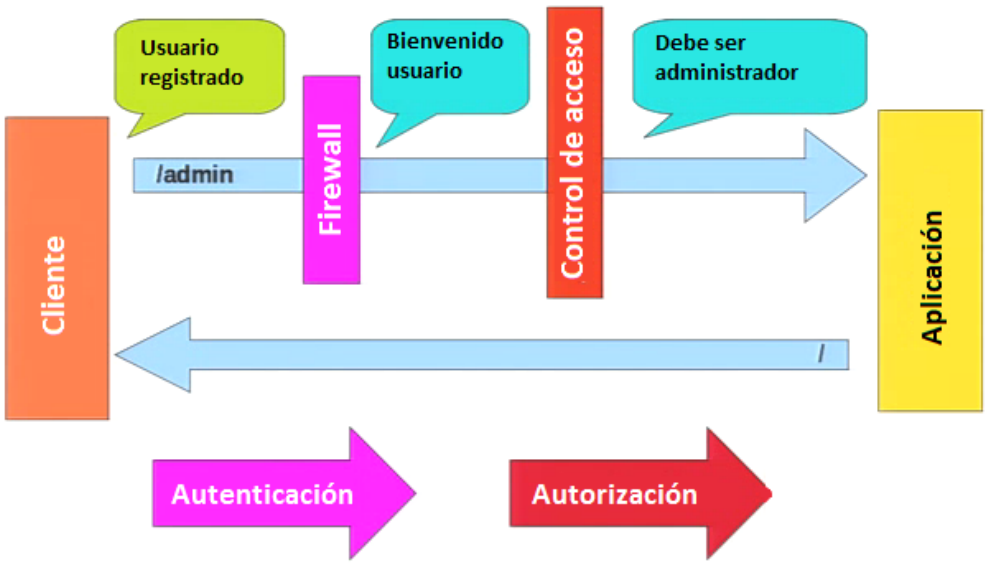


Figura 7. Sistema de seguridad.

3.2. Modelo de Diseño

El Modelo de Diseño es una abstracción del Modelo de Implementación y su código fuente, el cual fundamentalmente se emplea para representar y documentar su diseño. Es un modelo físico y concreto. Es usado como entrada esencial en las actividades relacionadas a implementación. Representa a los casos de uso en el dominio de la solución (37).

Diagrama de Clases del Diseño

El diagrama de clases es el diagrama principal de diseño y análisis para un sistema. Estos diagramas son ampliamente utilizados en el modelado de sistemas orientados a objetos, empleándose para representar las relaciones que se establecen entre las clases. Un diagrama de clases de diseño describe gráficamente las especificaciones de las clases de software y de las interfaces de la aplicación. Durante el análisis del sistema, el diagrama se desarrolla buscando una solución ideal. Durante el diseño, se usa el mismo diagrama, y se modifica para satisfacer los detalles de las implementaciones (38).

A continuación se muestra el diagrama de clases del diseño Web del caso de uso. Configurar Examen.

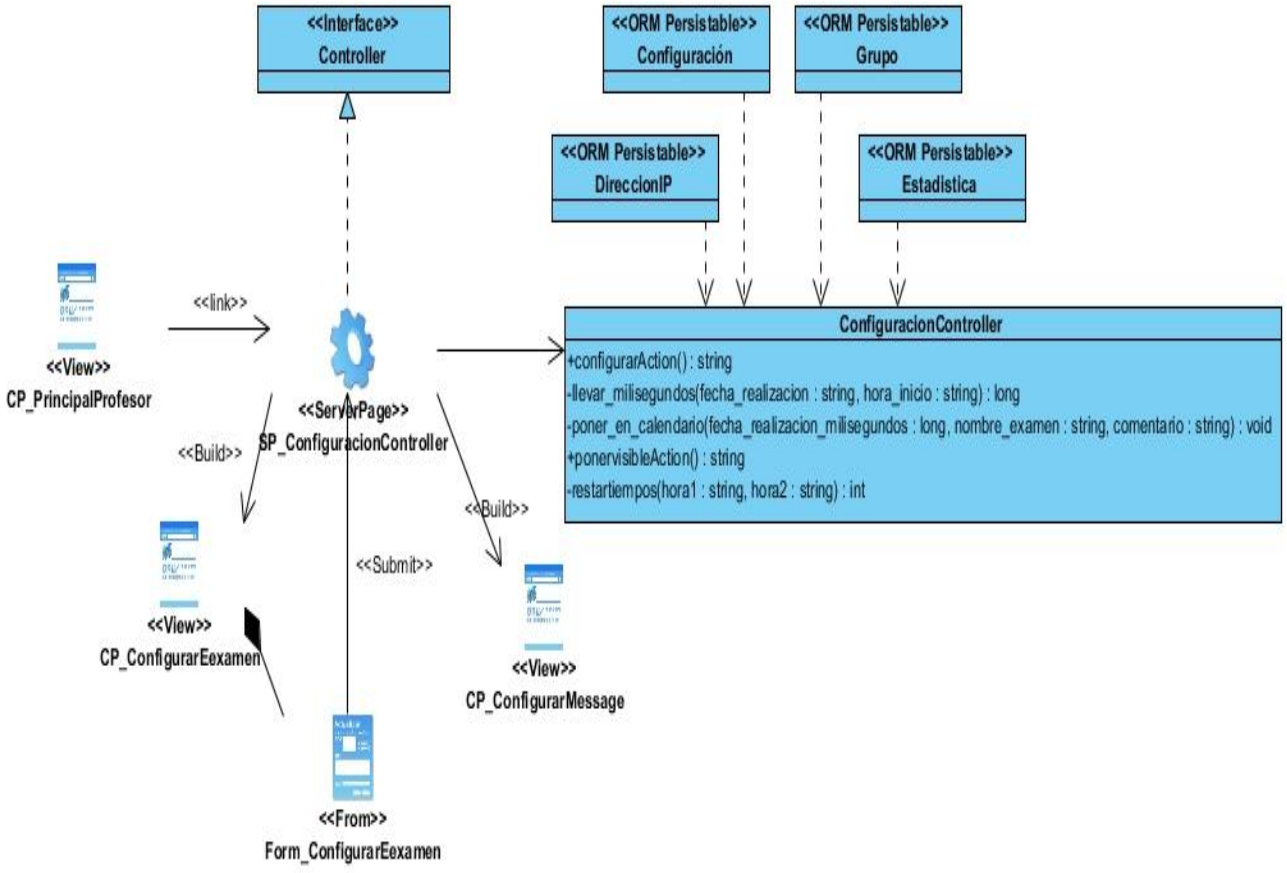


Figura 10. Diagrama de Clases del Diseño del Caso de Uso Configurar Examen.

A continuación se explican los patrones de diseño utilizados en el desarrollo del sistema.

3.3. Patrones de Diseño

¿Qué es un Patrón de Diseño?

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. Brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares (39).

Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias (40).

Los patrones de diseño se dividen en tres grupos principales (41).

Patrones de creación: Patrón de Fábrica Abstracta, Patrón Constructor, Patrón del Método de Fabricación, Patrón Prototipo, Patrón de Instancia Única (Singleton).

Patrones estructurales: Patrón Adaptador, Patrón Puente, Patrón Compuesto, Patrón Decorador, Patrón de Fachada, Patrón de Peso Mosca, Patrón Apoderado.

Patrones funcionales: Patrón de Cadena de Responsabilidad, Patrón de Comando, Patrón Intérprete, Patrón Iterador, Patrón Mediador, Patrón Memento, Patrón Observador, Patrón de Estado, Patrón de Estrategia, Patrón del Método Plantilla, Patrón Visitante.

En el desarrollo del sistema se aplicaron patrones de asignación de responsabilidades (*GRASP*) y patrones *Gang of Four* (*GOF*), con lo cual se aprovechan las características que brinda el *framework* de desarrollo Symfony2. Estos patrones son descritos a continuación.

3.3.1. Patrones GRASP

GRASP es un acrónimo que significa *General Responsibility Asignment Software Patterns* (Patrones Generales de Software para la Asignación de Responsabilidades).

Los patrones *GRASP* describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades (42). Los mismos constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objeto esencial y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable.

Experto: El uso de este patrón permite a los objetos valerse de su propia información para hacer lo que se les pide, favorece la existencia de mínimas relaciones entre las

clases, lo que permite contar con un sistema sólido y fácil de mantener. En el caso de Symfony2, las clases controladoras manejan las peticiones del cliente, y las clases (Entidades del negocio) son las empleadas en el acceso a datos, pues contienen y representan los datos que manejará el sistema, lo que permite que se conserve el encapsulamiento y se dé soporte a un bajo acoplamiento y una alta cohesión. Estas clases son mapeadas por el ORM Doctrine2, el cual se encarga de traducirlas a tablas en una base de datos y viceversa.

Controlador: Este patrón propone asignar la responsabilidad de controlar el flujo de eventos de un sistema, a clases específicas llamadas controladores. Los controladores no ejecutan las tareas sino que las delegan en otras clases, con las que mantiene un modelo de alta cohesión. Symfony2 contribuye a la utilización de este patrón debido a que define un Controlador Frontal (*Dispatcher*) que implica que todas las solicitudes son dirigidas a un script PHP que se encarga de instanciar al controlador frontal y redirigir las llamadas.

Creador: Se emplea en la asignación de responsabilidades a las clases relacionadas con la creación de objetos, de forma tal que una instancia de un objeto sólo pueda ser creada por el objeto que contiene la información necesaria para ello. El uso de este patrón admite crear las dependencias mínimas necesarias entre las clases, lo que beneficia el mantenimiento del sistema y se brindan oportunidades de reutilización. En el *framework* Symfony2 el controlador frontal (*Dispatcher*), es el encargado de instanciar las clases controladoras y éstas, a su vez, instancian objetos del modelo que son tratados por el ORM Doctrine2 mediante la utilización del patrón *Data Mapper.*

Bajo Acoplamiento: El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y recurre a ellas. El Bajo Acoplamiento soporta un diseño de clases más independientes, que reducen el impacto de cambios, y permite que sean más reutilizables. El uso de los patrones Experto y Creador favorecen al bajo acoplamiento entre las clases del sistema.

Alta Cohesión: Es el responsable de asignar una responsabilidad de forma tal que la cohesión siga siendo alta, este patrón caracteriza las clases que están estrechamente relacionadas y consiste en colaborar con otros objetos para compartir el esfuerzo si la tarea a realizar es grande. Cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos, y auto-identificable. Una clase con baja cohesión hace muchas cosas no relacionadas o hace demasiado trabajo.

3.3.2. Patrones Estructurales

Los patrones estructurales son utilizados por todo el *framework* y el ORM, estos son utilizados en todo tipo de situaciones.

Adapter (Adaptador): Este patrón adapta una interfaz para que pueda ser utilizada por una clase que de otro modo no podría utilizarla.

Facade (Fachada): Este patrón provee de una interfaz unificada simple para acceder a una interfaz o grupo de interfaces de un subsistema.

3.3.3. Otros patrones

Front-Controller: Este patrón propone utilizar un controlador como el punto inicial de contacto para manejar las peticiones del usuario en una aplicación. El controlador maneja el control de peticiones, incluyendo la invocación de los servicios de seguridad como la autentificación y autorización, la elección de una vista apropiada, el manejo de errores, y el control de la selección de estrategias de creación de contenido. La principal función del *Front Controller* es la de ser el único punto de entrada de peticiones, desde el cual se deriva luego al controlador específico que corresponda, según la dirección (URL) accedida. Se decide a que controlador derivar la petición según la dirección recibida. En el caso de Symfony2, las rutas definen que direcciones apuntan a que controlador. Este es el típico punto de entrada de un patrón MVC.

Una de las principales ventajas de utilizar un controlador frontal es que ofrece un punto de entrada único para toda la aplicación. Así, en caso de que sea necesario impedir el acceso a la aplicación, solamente es necesario editar el script correspondiente al controlador frontal. Si la aplicación no dispone de controlador frontal, se debería modificar cada uno de los controladores. A continuación se muestra una imagen que representa el flujo de trabajo de Symfony2 usando el controlador frontal.

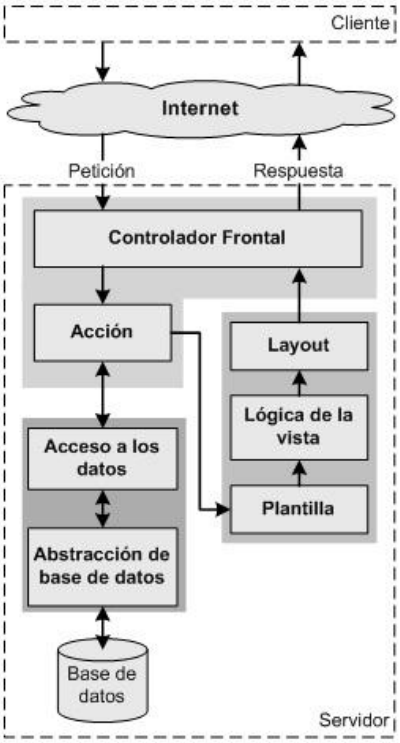


Figura 20. Flujo de trabajo de Symfony2 haciendo uso del Front-Controller.

Inyección de dependencias: Este es un patrón de diseño orientado a objetos, en el que se suministran objetos a una clase en lugar de ser la propia clase quien cree el objeto. Este consiste en resolver las dependencias de cada clase (atributos) generando los objetos cuando se inicia la aplicación y luego inyectarlos en los demás objetos que los necesiten a través de métodos set o bien a través del constructor, pero estos objetos se instancian una vez, se guardan en una factoría y se comparten por todos los usuarios. En la actualidad hay dos formas de extender Symfony2, una es haciendo uso de extensiones oficiales como son los *Vendor*, los cuales se pueden descargar e instalar, y otra es programando extensiones propias, lo cual se utiliza para suplir las extensiones que no se puedan encontrar. Estas extensiones se configuran en Symfony2 mediante etiquetas de Inyección de dependencias. Básicamente lo que se le da a conocer al *framework* es que utilice esos servicios de una forma especial. En el caso de twig se indica que el servicio que se va a hacer o implementar no va a ser un servicio más, sino que va a extender de un servicio ya existente que es twig.

Data Mapper: Este patrón pertenece a la capa de persistencia. Capa *mapper*, actúa como nexo entre objetos en memoria y su representación en la base de datos, siendo responsable de la transferencia de datos entre ambos lugares (43). Este patrón es implementado en el ORM Doctrine2, el cual es utilizado por Symfony2 para la gestión de sus datos.

3.4. Diagramas de Interacción

Los diagramas de Interacción representan como se comunican los objetos en una interacción. Los objetos interactúan para realizar colectivamente los servicios ofrecidos por las aplicaciones. Existen dos tipos de diagramas de interacción, los diagramas de colaboración y los diagramas de secuencia (44). Los elementos que componen los diagramas de interacción son los objetos y los mensajes.

Un Diagrama de Secuencia muestra la secuencia cronológica de mensajes entre objetos durante un escenario concreto. Están bien adaptados para representar interacciones. En este tipo de diagrama cada objeto viene dado por una barra vertical. El tiempo transcurre de arriba hacia abajo, y cuando existe demora entre el envío y la atención, se puede indicar usando una línea oblicua. A continuación se muestra un ejemplo de diagrama de secuencia.

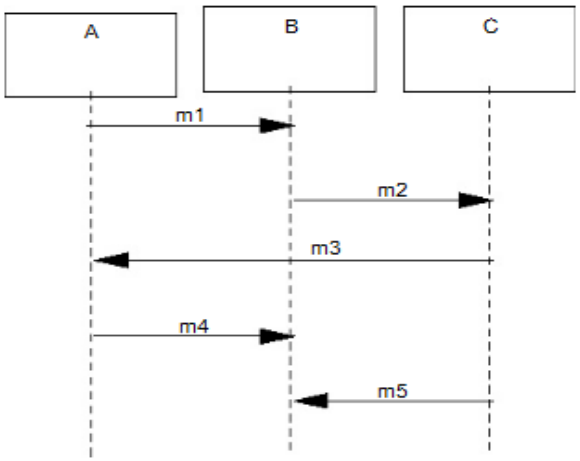


Figura 21. Representación de Diagrama de Secuencia.

Un diagrama de colaboración representa una colaboración, que es un conjunto de funciones de objeto relacionadas en un contexto determinado, y una interacción, que es un conjunto de mensajes intercambiados entre los objetos para lograr una operación o resultado determinado. Se trata de un diagrama de interacción que muestra cómo colaboran entre ellos un grupo de objetos, para un evento de sistema definido por un caso de uso.

A diferencia de un diagrama de secuencia, un diagrama de colaboración muestra relaciones entre funciones de objeto y no expresa tiempo como una dimensión independiente. Por tanto, los mensajes de un diagrama de colaboración se numeran para indicar su secuencia (45).

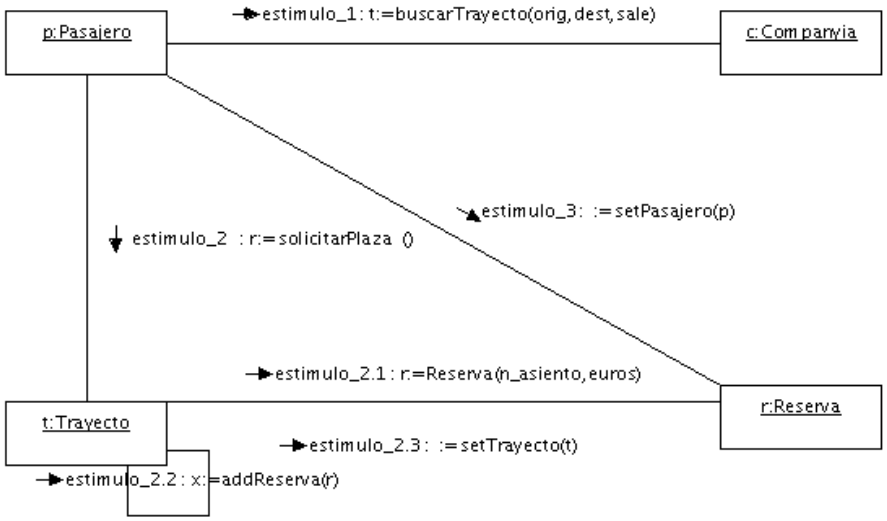


Figura 22. Representación de Diagrama de Colaboración.

A continuación se representa el flujo de eventos del caso de uso del sistema Gestionar Examen en diagramas de secuencias y colaboración.

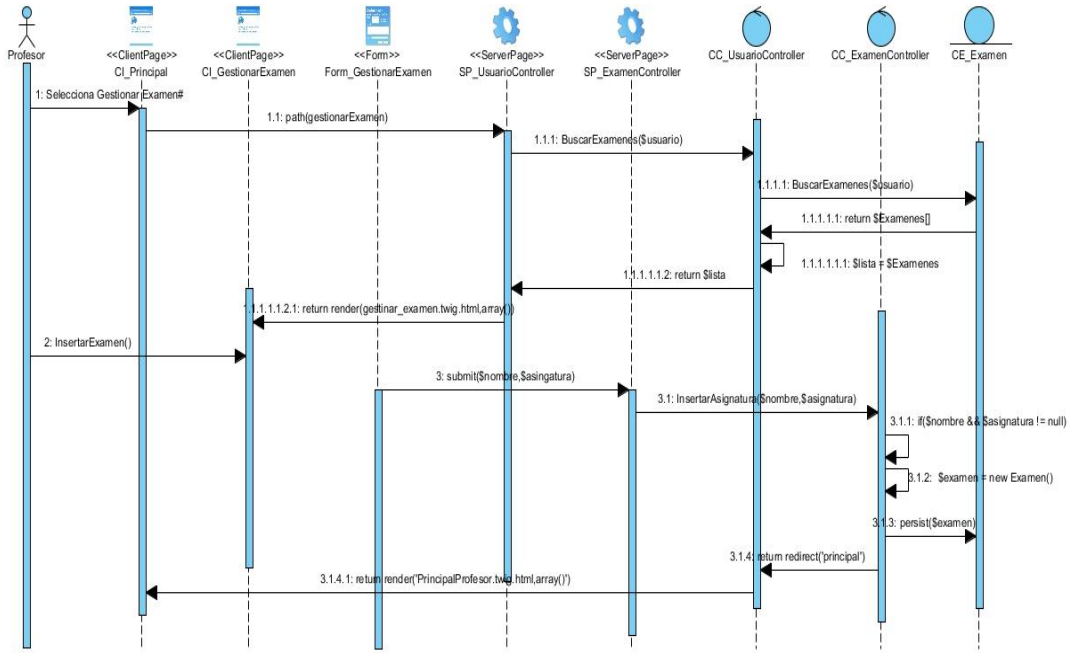


Figura 25. Diagrama de Secuencia del Caso de Uso Gestionar Examen.

A continuación se muestra el diagrama de Colaboración perteneciente al Caso de Uso (CU) crítico Gestionar Examen.

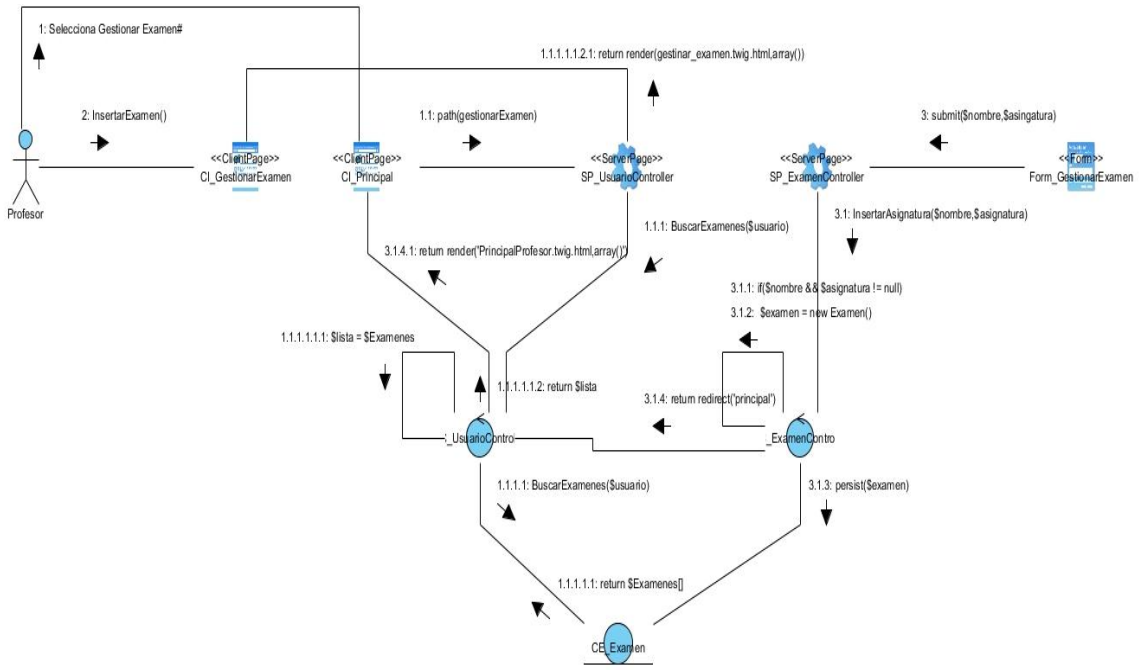


Figura 37. Diagrama de Colaboración del Caso de Uso Gestionar Examen.

3.5. Modelo de Datos

El Modelo de Datos describe las representaciones lógicas y físicas de datos persistentes utilizados por una aplicación. No son elementos físicos, sino abstracciones que permiten la implementación de un sistema eficiente de base de datos (46).

3.5.1 Modelo físico de datos (Modelo de Datos).

El Modelo de Datos representa la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos, así como la base formal para las herramientas y técnicas empleadas en el desarrollo y uso de sistemas de información. Es la estructura o representación física de las tablas de la Base de Datos. A continuación se muestra el modelo físico de datos.

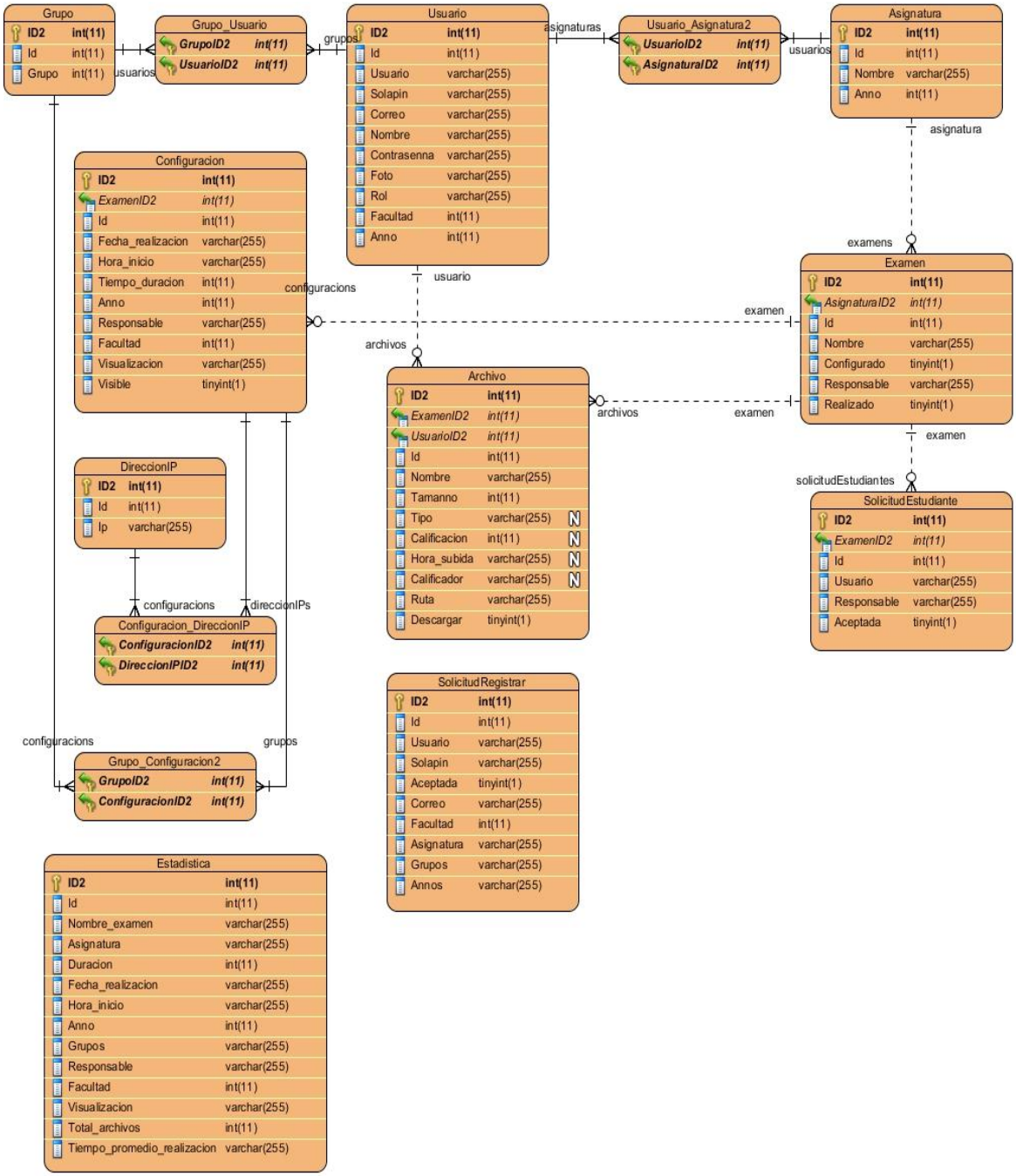


Figura 48. Diagrama del Modelo Físico de Datos.

Conclusiones

En el presente capítulo se llevó a cabo un estudio relacionado a la fase de diseño del sistema, el cual contempla los patrones de diseño y estilos arquitectónicos que benefician el buen desempeño de la próxima fase. También se describieron las representaciones lógicas y físicas de datos persistentes utilizados por el sistema, lo que permite avanzar a la próxima etapa poseyendo los conocimientos necesarios para su correcto desarrollo.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

En el presente capítulo se representarán un conjunto de diagramas que reflejan los componentes por los que está constituido el sistema, así como la distribución física de los mismos facilitando la comprensión de la estructura de la aplicación. También se diseñan los casos de prueba que permitirán evaluar y valorar la calidad del producto.

4.1. Modelo de Implementación.

En el modelo de implementación se describe cómo los elementos del modelo de diseño, como las clases, se deben implementar en términos de componentes. También cómo se organizan estos de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y en el lenguaje de programación utilizado. Además de la dependencia entre los componentes. Esto se hace a través del Diagrama de Despliegue y los Diagramas de Componentes (47).

4.1.1. Diagrama de Componentes.

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Los componentes representan todos los tipos de elementos software que entran en la fabricación de aplicaciones informáticas. Pueden ser archivos, paquetes de clases, bibliotecas dinámicas, entre otros elementos. Las relaciones de dependencia se utilizan en los diagramas de componentes para indicar que un componente utiliza los servicios ofrecidos por otro componente (48).

A continuación se realiza una breve descripción del diagrama de componentes de los casos de uso críticos del sistema Autenticar, Configurar Examen y Gestionar Examen.

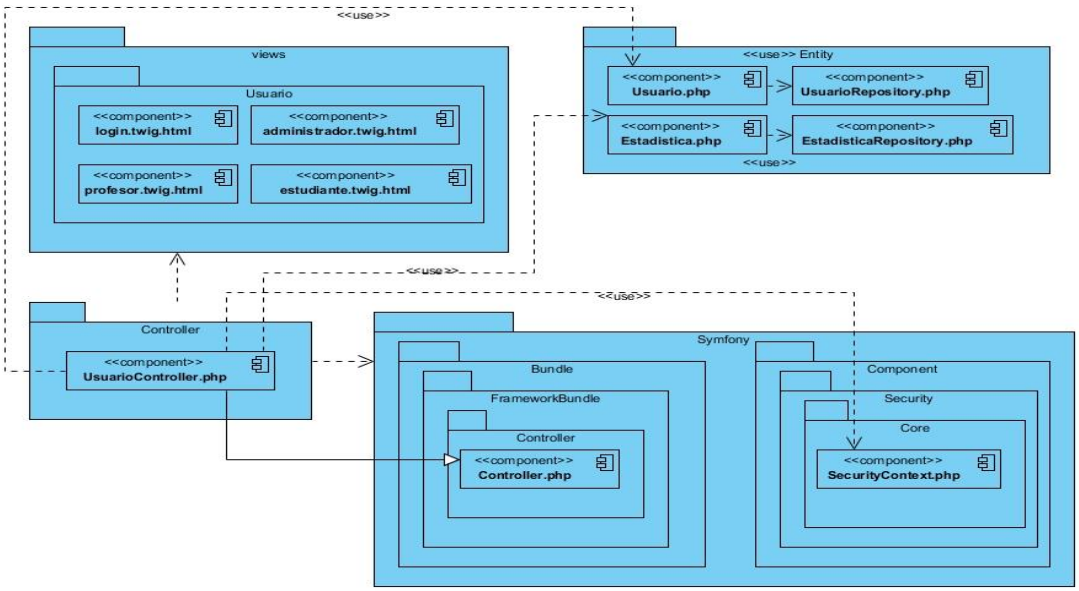


Figura 50. Diagrama de Componentes del Caso de Uso Autenticar.

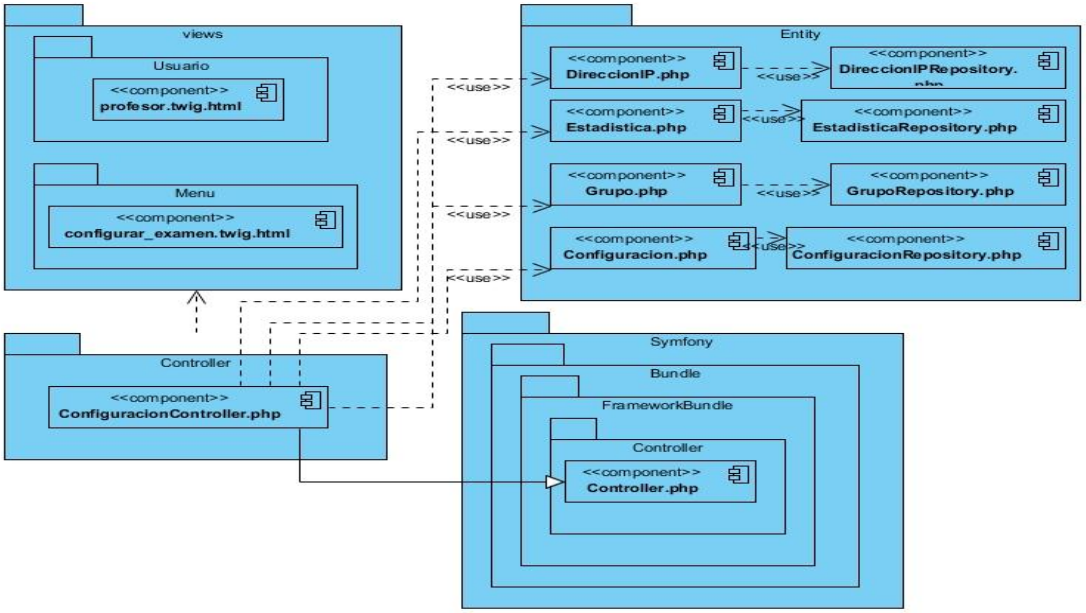


Figura 51. Diagrama de Componentes del Caso de Uso Configurar Examen.

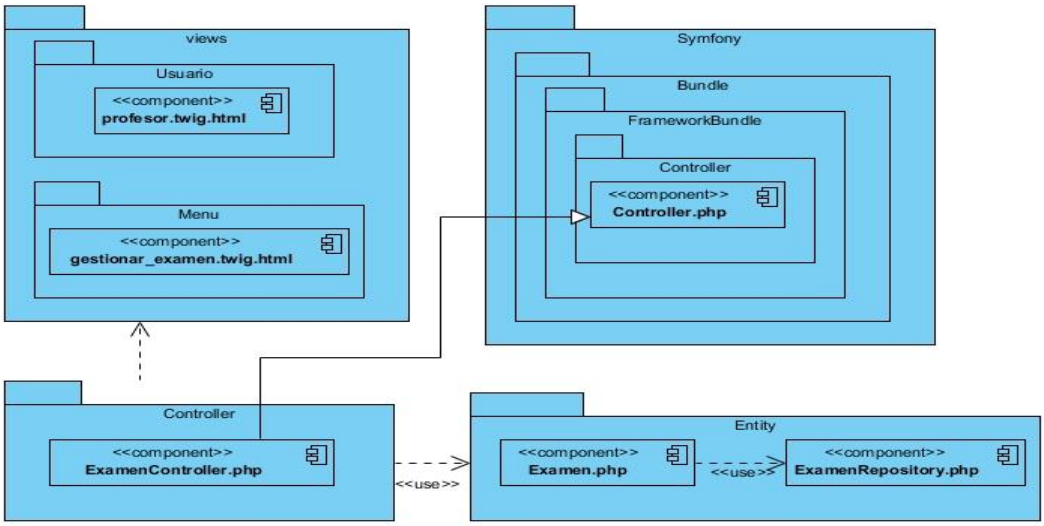


Figura 52. Diagrama de Componentes del Caso de Uso Gestionar Examen.

Los diagramas de componentes del sistema se realizaron utilizando la estructura ofrecida por el patrón Modelo-Vista-Controlador que propone Symfony2. De esta manera, se posibilitó ubicar adecuadamente los componentes dentro del paquete de las vistas (paquete *views*), ya que aquí es donde se permite visualizar la información obtenida y se logra gracias a los componentes ubicados dentro del paquete de las entidades (*Entity*) a través de los componentes situados en el paquete de controladores (*Controller*).

4.1.2. Diagrama de Despliegue.

El modelo de despliegue define la arquitectura física del sistema por medio de nodos interconectados. Estos nodos son elementos hardware sobre los cuales pueden

ejecutarse los elementos software. Con frecuencia se conoce como será la arquitectura física del sistema antes de comenzar su desarrollo. Por tanto se puede modelar los nodos y las conexiones del modelo de despliegue tan pronto como comience el flujo de trabajo de los requisitos (42).

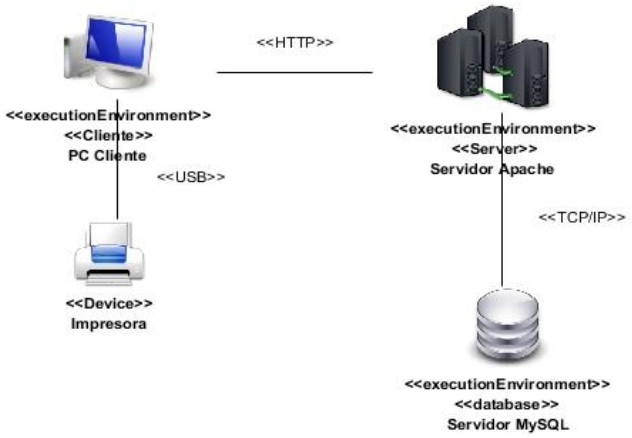


Figura 62. Diagrama de Despliegue del sistema.

4.2. Modelo de Prueba.

El modelo de pruebas describe como se prueban los componentes ejecutables en el modelo de implementación con pruebas de integración y de sistema. El modelo de pruebas puede describir también como han de ser probados aspectos específicos del sistema, como por ejemplo, si la interfaz de usuario es utilizable y consistente o si el manual de usuario del sistema cumple con su cometido. El modelo de pruebas es una colección de casos de prueba, procedimientos de prueba y componentes de prueba (42).

Un sistema debe ser sometido a diversas pruebas que deben ser llevadas a cabo tanto por el equipo de desarrolladores, como por los usuarios, equipos de operación y mantenimiento en la implantación, aceptación y mantenimiento del sistema de información. Dichas pruebas deben permitir validar y verificar la calidad del sistema en cuestión, además de identificar errores en la implementación, calidad y usabilidad del mismo.

4.2.1 Métodos de Prueba.

Actualmente existen dos métodos fundamentales de pruebas de software, estos son conocidos como el método de prueba de Caja Negra y el método de prueba de Caja Blanca.

Prueba de Caja Blanca.

Las pruebas de caja blanca comprueban los caminos lógicos del software. Además permiten examinar el estado del programa en varios puntos para determinar si el estado real coincide con el esperado. Dicha prueba es realizable sobre el código fuente del sistema que se decida probar.

Prueba de Caja Negra.

Las pruebas de caja negra, también denominada prueba de comportamiento, son diseñadas para validar los requisitos funcionales sin fijarse en el funcionamiento interno de un programa. Las técnicas de prueba de caja negra se centran en el ámbito de información de un programa, de forma que se proporcione una cobertura completa de prueba (49).

La prueba de caja negra intenta encontrar errores de las siguientes categorías:

- Funciones incorrecta o ausente.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a base de datos externas.
- Errores de rendimiento.
- Errores de inicialización y determinación.

Tras el desarrollo de la aplicación se realizaron dos pruebas fundamentales, estas fueron las pruebas de funcionalidades, las cuales se basaron en casos de pruebas de requisitos funcionales y las pruebas de carga y estrés, las cuales fueron necesarias para tener conocimiento del rendimiento del sistema en condiciones extremas.

4.2.2 Prueba de Funcionalidades.

Los casos de prueba que se utilizarán para las pruebas de la aplicación, están diseñados para verificar los requerimientos del usuario. A continuación se presentan los casos de pruebas realizados a las principales funcionalidades del sistema, siguiendo el método de caja negra y su técnica Partición de Equivalencia. Esta técnica es una de las más efectivas en este método de prueba ya que permite examinar los valores válidos y no válidos de las entradas existentes en el software y revela de forma inmediata los errores que de otro modo requerirían una gran cantidad de ejecuciones del sistema antes de ser detectados.

Para llevar a cabo el desarrollo de esta técnica se realizó una división del campo de entrada seleccionado, en variables de equivalencia con datos de entrada y salida. Dichas variables se clasifican en dos tipos: válidas y no válidas. Las variables válidas

representan entradas válidas al programa, mientras que las no válidas representan valores de entrada erróneos.

Escenario 1: Caso de Prueba del Requisito Funcional Insertar Examen.

Descripción: El usuario selecciona la opción Gestionar exámenes.

Condiciones: Debe haberse autenticado en el sistema como profesor.

Tabla 20. Caso de Prueba del Requisito Funcional Insertar Examen.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 3.1 Crear examen con datos válidos	Se crea el examen correctamente en el sistema.	El sistema redirecciona al profesor a su página principal.	Se selecciona en el menú principal la opción Gestionar exámenes, se rellena el formulario y se oprime el botón “Crear examen”.
EC 3.2 Crear examen con datos no válidos en el campo “Nombre del examen”	No se crea el examen en el sistema y se muestra un mensaje de error.	El sistema muestra el mensaje: “*El campo del nombre no debe contener caracteres especiales*”, luego vacía el campo “Nombre del examen”.	Se selecciona en el menú principal la opción Gestionar exámenes, se rellena el formulario y se oprime el botón “Crear examen”.
EC 3.3 Crear examen con campos en blanco	No se crea el examen en el sistema y se muestra un mensaje de error.	El sistema muestra el mensaje: “No deben existir campos en blanco”.	Se selecciona en el menú principal la opción Gestionar exámenes, se rellena el formulario y se oprime el botón “Crear examen”.

Escenario 2: Caso de Prueba del Requisito Funcional Modificar Examen.

Descripción: El usuario selecciona la opción Gestionar exámenes.

Condiciones: Debe haberse autenticado en el sistema como profesor.

Tabla 21. Caso de Prueba del Requisito Funcional Modificar Examen.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 4.1 Modificar examen con datos válidos	Se modifica el examen correctamente en el sistema.	El sistema redirecciona al profesor a su página principal.	Se selecciona en el menú principal la opción Gestionar exámenes, se rellena el formulario y se oprime el botón "Modificar examen seleccionado".
EC 4.2 Modificar un examen con datos no válidos en el campo "nuevo nombre"	No se modifica el examen en el sistema y se muestra un mensaje de error.	El sistema muestra el mensaje: "*El campo nuevo nombre no debe contener caracteres especiales*", luego vacía el campo "nuevo nombre".	Se selecciona en el menú principal la opción Gestionar exámenes, se rellena el formulario y se oprime el botón "Modificar examen seleccionado".
EC 4.3 Modificar examen con campos en blanco	No se modifica el examen en el sistema y se muestra un mensaje de error.	El sistema muestra el mensaje: "No deben existir campos en blanco".	Se selecciona en el menú principal la opción Gestionar exámenes, se rellena el formulario y se oprime el botón

			“Modificar examen seleccionado”.

Escenario 3: Caso de Prueba del Requisito Funcional Configurar Examen.

Descripción: El usuario selecciona la opción Configurar examen.

Condiciones: Debe haberse autenticado en el sistema como profesor.

Tabla 22. Caso de Prueba del Requisito Funcional Configurar Examen.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 5.1 Configurar examen con datos válidos.	Se configura el examen correctamente en el sistema.	El sistema redirecciona al profesor a su página principal.	Se selecciona en el menú principal la opción Configurar examen, se rellena el formulario y se oprime el botón “Aplicar configuración”.
EC 5.2 Configurar examen con datos no válidos en el campo “Fecha de realización”.	No se configura el examen y el sistema muestra un mensaje de error.	El sistema muestra el mensaje: “por favor verifique el campo de la fecha”.	Se selecciona en el menú principal la opción Configurar examen, se rellena el formulario y se oprime el botón “Aplicar configuración”.
EC 5.3 Configurar examen con datos no válidos en el	No se configura el examen y el sistema muestra un mensaje de	El sistema muestra el mensaje: “por favor verifique la duración del examen”.	Se selecciona en el menú principal la opción Configurar examen, se

campo “Hora”.	error.		rellena el formulario y se oprime el botón “Aplicar configuración”.
EC 5.4 Configurar examen con datos inválidos en el campo “Min”.	No se configura el examen y el sistema muestra un mensaje de error.	El sistema muestra el mensaje: “por favor verifique la duración del examen”.	Se selecciona en el menú principal la opción Configurar examen, se rellena el formulario y se oprime el botón “Aplicar configuración”.
EC 5.5 Configurar un examen con el campo “Comentario” en blanco.	Se configura el examen correctamente en el sistema.	El sistema muestra el mensaje: “No ha insertado un comentario!, desea aplicar la configuración al examen sin comentario”. El sistema redirecciona al profesor a su página principal.	Se selecciona en el menú principal la opción Configurar examen, se rellena el formulario y se oprime el botón “Aplicar configuración”.

Resultados de las pruebas de funcionalidades.

Las pruebas se realizaron en una primera, segunda, tercera y cuarta iteración. La figura que se muestra a continuación muestra que en la primera iteración se detectaron 15 no conformidades de las cuales fueron resueltas 14, en la segunda iteración se detectaron 4 no conformidades incluyendo la no conformidad que quedó de la primera iteración, de las cuales las 4 fueron resueltas, en la tercera iteración se detectaron 2 no conformidades, resolviéndose las 2 respectivamente y en la cuarta iteración no se detectaron no conformidades. Las no conformidades detectadas fueron resueltas en el menor tiempo posible por el desarrollador, exceptuando 1 de la primera iteración, la cual no pudo ser solucionada en el momento, quedando pendiente para las otras iteraciones.

La solución de las no conformidades contribuyó a que la aplicación aumentara su calidad, y pudiera cumplir con los requisitos funcionales propuestos.

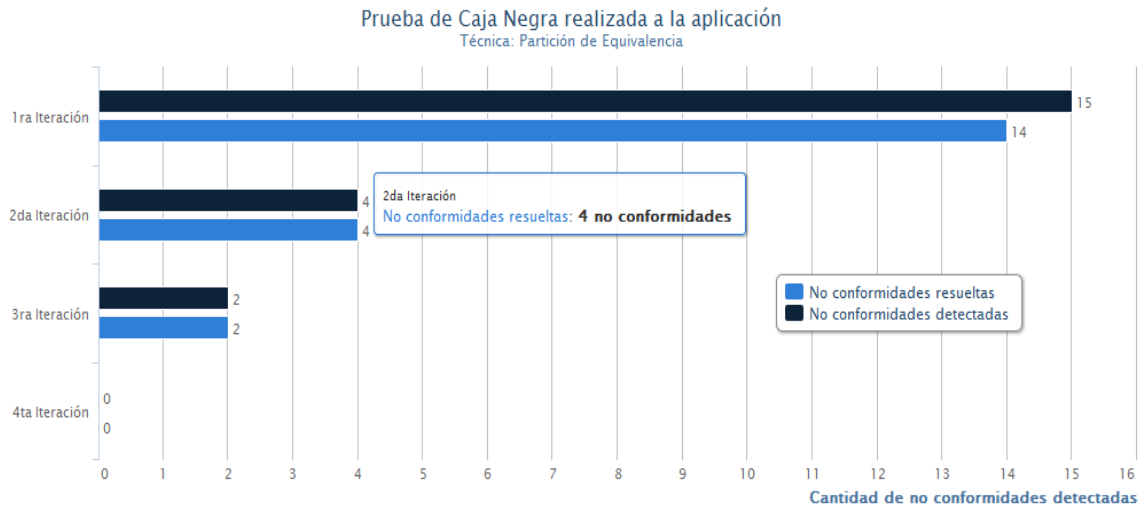


Figura 62. Gráfica de No Conformidades.

4.2.3 Pruebas de carga y estrés.

Dada la situación problemática por la que fue construido el sistema en cuestión, se hizo necesario realizar pruebas de carga y estrés. Dicha prueba pone al sistema en un ambiente extremo, simulando una gran cantidad de conexiones simultáneas, igualando o sobrepasando a las realizadas en un escenario real.

La prueba se realizó utilizando la herramienta *JMeter* y contando con el apoyo de un hardware de 1 GB (Gigabytes) de memoria RAM, un Procesador Intel(R) Core(TM)2 Duo a 2.20 GHz (Gigahercio), y una capacidad de disco duro de 160 Gb. Para la realización de la prueba se tomó una muestra de 200 conexiones simultáneas, comparando y comprobando así el tiempo de respuesta del sistema y el rendimiento de las funcionalidades.

A continuación se muestran los resultados obtenidos para una cantidad de 200 conexiones simultáneas.

Label	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
Petición HTTP	200	25338	25497	44614	1403	48887	0,00%	4,0/sec	30,4
TOTAL	200	25338	25497	44614	1403	48887	0,00%	4,0/sec	30,4

Figura 63. Informe Agregado de la herramienta JMeter para 200 conexiones simultáneas.

Conclusiones

En este capítulo se describieron los elementos necesarios para la implementación, a través del diagrama de despliegue y el diagrama de componentes, quedando así conformado el modelo de implementación del sistema. Además, se realizan pruebas de funcionalidades y carga y estrés. En dichas pruebas se muestran los resultados de casos de pruebas definidos, que fueron aplicados a los requisitos funcionales del sistema, así como el tiempo de respuesta del sistema y el rendimiento de las funcionalidades ante una situación extrema.

Conclusiones Generales

Como resultado de la presente investigación se obtuvo una herramienta que permite configurar los exámenes realizados en los laboratorios, minimizando la posibilidad de fraude académico y agilizando el proceso de realización de exámenes. Además se desarrollaron cada una de las tareas que fueron plasmadas al comienzo de la investigación, dándole así cumplimiento al objetivo general propuesto referente a la problemática expuesta. Entre los resultados que la presente investigación arrojó se encuentran:

- Se efectuó un estudio del proceso de realización de exámenes en la universidad, abarcando los tipos de exámenes que se realizan, las herramientas utilizadas en la ejecución de los mismos, entre otros parámetros vinculados a dicho proceso.
- Se fundamentó la selección de la metodología, lenguaje de modelado, tecnologías y herramientas para el desarrollo del Sistema para la configuración de exámenes en los laboratorios.
- Se diseñó el negocio utilizando la notación gráfica estandarizada BPMN. Comprendiéndose de una mejor forma de los procesos llevados a cabo en la realización de los exámenes efectuados en los laboratorios. Además se definieron los requisitos funcionales y no funcionales de acuerdo a las necesidades del sistema, estableciendo de esta forma un consenso entre los desarrolladores y el cliente final.
- Se realizó el análisis y diseño del sistema, seleccionando como punto principal la arquitectura Cliente/Servidor, empleando el patrón arquitectónico Modelo Vista Controlador (MVC). Además se llevó a cabo el modelado del diseño del sistema mediante los diagramas de interacción, los diagramas de clases del diseño y modelo de datos. También quedaron plasmados los patrones de diseños empleados en el desarrollo del sistema.
- Fueron diseñados e implementados los componentes necesarios para el correcto funcionamiento del sistema. Estos permitieron organizar y definir las dependencias entre códigos fuentes y librerías que utilizará el sistema.
- Se realizaron pruebas de funcionalidades y de carga y estrés, las cuales demostraron el buen funcionamiento del sistema en distintos escenarios de pruebas, comprobando así su funcionamiento de acuerdo a los requisitos planteados, además de su rendimiento y tiempo de respuesta en situaciones extremas.

Recomendaciones

- Implementar las funcionalidades que permitan desconectar a un estudiante de la aplicación en tiempo real. Esta característica sería de importancia en el tema de los fraudes, ya que en caso de que algún examen fuese configurado para un año específico y accedan estudiantes no autorizados, el profesor responsable de la realización de dicho examen pueda tomar el control de la situación.
- Implementar las funcionalidades que permitan obtener la información de un determinado usuario (forma del directorio UCI). Esta funcionalidad se complementaría con la recomendación anterior, ya que se hace necesario conocer de forma precisa los datos más relevantes de un estudiante que comete o intenta cometer fraude.
- Implementar las funcionalidades que permita a un estudiante o grupo de estudiantes específicos, de diferentes años y grupos, realizar un examen comprobatorio.
- Implementar las funcionalidades que identifique cuando un usuario intenta subir un examen desde varias direcciones IP.
- Implementar las funcionalidades que permita la subida de los exámenes desde una determinada subred.

Bibliografía

1. **española, Diccionario de la lengua.** Real Academia Española. *Real Academia Española.* [En línea] Real Academia Española © Todos los derechos reservados. [Citado el: 1 de 6 de 2013.] http://lema.rae.es/drae/?val=superación.

2. Campus MVP. *Campus MVP.* [En línea] [Citado el: 8 de 1 de 2013.] http://www.campusmvp.com/catalogo.

3. Campus MVP. *Campus MVP.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.campusmvp.com/catalogo/Product-jQuery-paso-a-paso-para-programadores-ASP.NET_100.aspx.

4. **MsC. Ileana R. Alfonso Sánchez1.** Biblioteca Virutal en Salud. *Biblioteca Virutal en Salud.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.bvs.sld.cu/revistas/aci/vol11_1_03/aci02103.htm.

5. **Red telemática de salud en Cuba.** Infomed. *Infomed.* [En línea] Copyright© 1998 InfoMed, Red Telemática de Salud en Cuba. [Citado el: 1 de 6 de 2013.] http://www.sld.cu/libros/distancia/cap1.html.

6. **University of Maryland.** Infomed. *Infomed.* [En línea] Copyright© 1998 InfoMed, Red Telemática de Salud en Cuba. [Citado el: 1 de 6 de 2013.] http://www.sld.cu/libros/distancia/cap1.html.

7. **Distance Education at a Glance.** Infomed. *Infomed.* [En línea] Copyright© 1998 InfoMed, Red Telemática de Salud en Cuba. [Citado el: 1 de 6 de 2013.] http://www.sld.cu/libros/distancia/cap1.html.

8. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Moodle.

9. Tendenciaseducativas. *Tendenciaseducativas.* [En línea] @Centro Internacional de Tecnologías Avanzadas. FGSR. [Citado el: 1 de 6 de 2013.] http://www.tendenciaseducativas.es/index.php?option=com_content&view=article&id=66&Itemid=113..

10. Entorno Virtual de Aprendizaje (EVA). *Entorno Virtual de Aprendizaje (EVA).* [En línea] Universidad de las Ciencias Informáticas, 2013. [Citado el: 1 de 6 de 2013.] http://eva.uci.cu.

11. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Caroline.

12. Claroline. *Claroline.* [En línea] © 2012 Claroline. [Citado el: 1 de 6 de 2013.] http://www.claroline.net/?lang=es.

13. Scribd. *Scribd.* [En línea] © Copyright 2013 Scribd Inc. [Citado el: 1 de 6 de 2013.] http://es.scribd.com/doc/34158569/Comparacion-entre-distintas-plataformas-e-learning-Moodle-Dokeos-Caroline-ATutor-y-EFront.

14. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Redmine.

15. GESPRO. *GESPRO.* [En línea] [Citado el: 1 de 6 de 2013.] http://gespro-help.prod.uci.cu/.

16. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Blackboard.

17. **Servicios de Apoyo Computacinal / SAC.** Sistema Tecnológico de Monterrey. *Sistema Tecnológico de Monterrey.* [En línea] [Citado el: 1 de 6 de 2013.] www.cem.itesm.mx/consulta/modeloeducativo/blackboard/Manual_BB.doc.

18. Sakai. *Sakai.* [En línea] Sakai Foundation under the Creative Commons Attribution 3.0 United States License. [Citado el: 1 de 6 de 2013.] http://www.sakaiproject.org/es/acerca-de-sakai.

19. **Mtra. María de Lourdes Santiago Zaragoza.** UTVM. *UTVM.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.utvm.edu.mx/OrganoInformativo/orgJul07/RUP.htm.

20. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Visual_Paradigm.

21. **Miguel Angel Alvarez.** DesarrolloWeb.com. *DesarrolloWeb.com.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.desarrolloweb.com/articulos/392.php.

22. **María Jesús Lamarca Lapuente.** HiperTexto. *HiperTexto.* [En línea] Universidad Complutense de Madrid. Facultad de Ciencias de la Información. Dpto. de Biblioteconomía y Documentación. [Citado el: 1 de 6 de 2013.] http://www.hipertexto.info/documentos/html.htm.

23. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/JavaScript.

24. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Mysql.

25. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Servidor_Apache.

26. Maestros del Web. *Maestros del Web.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.maestrosdelweb.com/editorial/comparacion-frameworks-javascript.

27. Hoplite. *Hoplite.* [En línea] © 2005 - 2013 Hoplite Software Derechos Reservados. Hoplite es una marca registrada de Hoplite Software, S.L. Hoplite Software, S.L. [Citado el: 1 de 6 de 2013.] http://www.hoplite.es/tecnologias/symfony.

28. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Requisitos_de_Software.

29. visual-paradigm.com. *visual-paradigm.com.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.visual-paradigm.com/product/vpuml/tutorials/writingeffectiveusecase.jsp.

30. **Carlos Billy Reynoso.** carlosreynoso.com.ar. *carlosreynoso.com.ar.* [En línea] [Citado el: 1 de 6 de 2013.] http://carlosreynoso.com.ar/archivos/arquitectura/Introduccion.PDF.

31. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Estilos_arquitectónicos.

32. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Patrones_de_diseño_y_arquitectura.

33. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Cliente-Servidor#Arquitectura_Cliente-Servidor.

34. Revista Digital de las Tecnologías de la Información y las Comunicaciones. *Revista Digital de las Tecnologías de la Información y las Comunicaciones.* [En línea] [Citado el: 1 de 6 de 2013.] http://revistatelematica.cujae.edu.cu/index.php/tele/article/download/15/10.

35. LobrosWeb. *LobrosWeb.* [En línea] © 2013 LibrosWeb.es. [Citado el: 1 de 6 de 2013.] http://librosweb.es/symfony_1_2/capitulo_2/el_patron_mvc.html.

36. **Nacho Pacheco.** gitnacho.github.io. *gitnacho.github.io.* [En línea] © Copyright 2011-2013. [Citado el: 1 de 6 de 2013.] http://gitnacho.github.io/symfony-docs-es/book/security.html.

37. MeRinde. *MeRinde.* [En línea] Copyright © 2013 MeRinde V1.1.0 . [Citado el: 1 de 6 de 2013.] http://merinde.net/index.php?option=com_content&task=view&id=92&Itemid=296.

38. GeofísicaUnam. *GeofísicaUnam.* [En línea] [Citado el: 1 de 6 de 2013.] http://mmc.geofisica.unam.mx/LuCAS/Tutoriales/doc-modelado-sistemas-UML/multiple-html/x219.html.

39. **Nicolás Tedeschi.** MSDN. *MSDN.* [En línea] [Citado el: 1 de 6 de 2013.] http://msdn.microsoft.com/es-es/library/bb972240.aspx.

40. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Patón_de_diseño_de_software.

41. Kioskea.net. *Kioskea.net.* [En línea] [Citado el: 1 de 6 de 2013.] http://es.kioskea.net/contents/224-patrones-de-diseno.

42. **Booch, Grady, Jacobson, Ivar y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* s.l. : PEARSON EDUCACIÓN, S.A, 2000. 84-7829-036-2.

43. Scribd. *Scribd.* [En línea] [Citado el: 1 de 6 de 2013.] http://es.scribd.com/doc/54430680/101/Datamapper.

44. Scribd. *Scribd.* [En línea] [Citado el: 1 de 6 de 2013.] http://es.scribd.com/doc/58905500/Diagramas-de-Interaccion-UML.

45. Microsoft. *Microsoft.* [En línea] © 2013 Microsoft Corporation. Todos los derechos reservados. [Citado el: 1 de 6 de 2013.] http://office.microsoft.com/es-es/visio-help/diagramas-de-colaboracion-de-uml-HP001208811.aspx?CTT=1.

46. Ecured. *Ecured.* [En línea] [Citado el: 1 de 6 de 2013.] http://www.ecured.cu/index.php/Bases_de_datos#Modelos_de_bases_de_datos.

47. Reminde.net. *Reminde.net.* [En línea] [Citado el: 1 de 6 de 2013.] http://merinde.net/index.php?option=com_content&task=view&id=495&Itemid=291.

48. **GARCIA SAAVEDRA MADELINE TRACY, GONZALES SOTO CLAUDIA AURELIA, SIERRA ESTEVEZ PAOLA VANINA, YAPUCHURA VASQUEZ GREDTZEL MARIEL.** virtual.usalesiana.edu.bo. *virtual.usalesiana.edu.bo.* [En línea] [Citado el: 1 de 6 de 2013.] virtual.usalesiana.edu.bo/web/practica/archiv/componn.doc.

49. **Pressman, Roger S.** *Ingeniería del software: Un Enfoque Práctico.* s.l. : McGraw-Hill, 2002. 8448132149.