Universidad de las Ciencias Informáticas

Facultad 15

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas.

Titulo: Análisis y Diseño de una herramienta para planificar y controlar la ejecución de las evaluaciones prácticas docentes.

Autor(es): Angélica Viviana Cedeño Milanés.

Sicelis Barnes Hinojosa.

Tutor(es): Ing. Maikel Pérez Martínez.

Ciudad de La Habana, Habana

"Año 52 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Angélica V. Cedeño Milanés _____

Sicelis Barnes Hinojosa _____

_____

Ing. Maikel Pérez Martínez

DATOS DE CONTACTO

Tutor: Ing.Maikel Pérez Martínez.

Breve Currículo:

Graduado de Ingeniero en Ciencias informáticas, en la Universidad de Ciencias Informáticas en el año 2007.Ha desempeñado los roles de diseñador y programador dentro del proyecto SIGEP desde el año 2008.

Correo Electrónico: mperezma@uci.cu

Años de graduado: 2

AGRADECIMIENTOS

A todos los que han hecho posible la realización de esta tesis. A mis padres Silvia y Edilberto y a mi hermanito Yasel por todo el esfuerzo y esmero.

A mi familia toda por su ayuda.

A mi novio por estar siempre presente, en el momento justo.

A mis amigos de siempre por apoyarme; a Julio, Leo, Yenia, Daymi y todos a los que considero mis amigos.,

A todos los que siempre han creído en mí y esperan ansiosos este momento

A nuestro tutor por sus consejos, por su ayuda, y por estar pendiente en el correcto desempeño de esta tesis.

Sicelis Barnes Hinojosa.

A mis padres por transmitirme siempre la idea y el apoyo en el momento preciso, por ayudarme a lidiar con cada una de mis dudas.

A mi hermano por no requerir de las palabras para saber qué sentimos y cuánto nos necesitamos. A toda mi familia, por poner más que un grano de arena para materializar este sueño compartido.

A mi novio por ser mi confidente, mi amigo, por estar siempre al tanto de mis problemas y saber que decirme en el momento preciso y por darme la posibilidad de conocer a personas tan especiales como lo son Dainelis, Juan, Marilín y Maria Tereza.

A los amigos y compañeros de estudio a los cuales agradezco por la compañía de estos años, por la satisfacción de estar juntos en tan diversas experiencias.

A la UCI por resultar tan positiva para mi formación, facilitándome crecimiento profesional y personal.

Angélica Viviana Cedeño Milanés.

DEDICATORIA

Dedico este trabajo a mis padres, a mi hermano y a mi novio por todo el apoyo que me han dado a lo largo de estos años, ya que sin ellos no hubiese podido llegar hasta aquí, a ellos les dedico también mi título y les estoy eternamente agradecida.

Sicelis Barnes Hinojosa.

Este trabajo ha sido posible entre otras cosas, al apoyo incalculable de mis padres, pues el esfuerzo y el ánimo empleado para desarrollar el mismo, se los debo a ellos, es por eso, que les dedico este trabajo y mi título con todo el cariño de este mundo. A mi hermano también va esta dedicatoria, pues es mi otra mano derecha.

Angélica Viviana Cedeño Milanés.

RESUMEN

La Universidad de las Ciencias Informáticas es una entidad nueva, llamada a jugar un papel rector en el proceso de informatización y digitalización del país.

En la misma se lleva a cabo la tarea de informatizar todos sus procesos, en la que se incluyen los procesos asociados con el perfeccionamiento del sistema evaluativo. Dando respuesta a estas necesidades se desarrolló este trabajo con el objetivo de: Realizar el Análisis y el Diseño de una herramienta para planificar y controlar la ejecución de las evaluaciones prácticas docentes convocados por el Departamento de Programación de la Facultad 4.

El trabajo está dividido en tres capítulos, que recogen la Fundamentación Teórica: análisis de aspectos necesarios para comprender los procesos en desarrollo, así como otras aplicaciones similares existentes a nivel internacional, nacional y de la universidad, así como las diferentes metodologías, herramientas y lenguajes de modelado a utilizar en el desarrollo del sistema; Descripción de la Solución, donde se estudia todo el modelado del negocio, modelado del sistema, requerimientos funcionales y no funcionales así como vistas y descripciones de casos de uso; se realiza además un Diseño de la Solución donde queda constituido el prototipo de interfaz, arquitectura del sistema, clases del sistema así como la vista del despliegue.

PALABRAS CLAVES

Administración del sistema, Administración de configuraciones, Administración de exámenes, Informes y Notificaciones, Análisis, Diseño.

ÍNDICE

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Introducción.

"Educar es depositar en cada hombre la obra humana que le ha antecedido: es hacer a cada hombre resumen del mundo viviente, hasta el día en que vive: es ponerlo a nivel de su tiempo, para que flote sobre él y no dejarlo debajo de su tiempo, con lo que no podrá salir a flote; es preparar al hombre para la vida".

José Martí.

En función de cumplir este precepto martiano, es necesario que la educación esté a la vanguardia en el desarrollo científico técnico de cada época histórica. Hoy, cuando el conocimiento y la información constituyen un recurso de poder, requieren que el hombre aprenda sus leyes para procesar información, adquirir conocimientos e interactuar efectivamente con su medio, tanto en el ámbito científico como económico y social. En la actualidad el proyecto educacional cubano ha priorizado la enseñanza de la computación como una tecnología necesaria para el progreso socio- económico, en tal sentido, la inclusión de la enseñanza en el sistema educacional es una realidad desde la década del 80. Hoy es imposible pensar en el acceso a la información y en la gestión del conocimiento sin el uso y la aplicación de las nuevas tecnologías de la información y la comunicación.

El desarrollo de Internet, y particularmente de herramientas cada vez más eficientes para la comunicación y la interacción humanas, posibilitan el aprendizaje activo y colaborador en red, con la creación de espacios virtuales que se integran a los procesos docentes de carácter presencial. Es una nueva dimensión formativa que está cambiando no solo los tradicionales medios de enseñanza en herramientas para el aprendizaje, sino lo más importante: la comunicación durante el proceso de enseñanza-aprendizaje y su propia dinámica.

Dominar la información necesaria y la posibilidad de elaborarla en forma adecuada ha devenido en nuestros días una de las características más importante del progreso social. Basado en lo expuesto anteriormente surge la idea de realizar una aplicación que permita de una forma u otra contribuir al perfeccionamiento de la educación en la rama de la informática así como a la explotación de todos los recursos con los que actualmente cuenta la Facultad 4 de la Universidad de las Ciencias Informáticas.

En la actualidad la universidad cuenta con los recursos, el conocimiento y los medios necesarios para perfeccionar el sistema educacional actual haciendo uso de las nuevas tecnologías, pero existen problemas con el aprovechamiento y la incorporación de las nuevas técnicas de cómputo y procesamiento de la información. Hoy por hoy, cuenta con un sitio Web que no responde a las necesidades que hoy se enfrentan .Se necesita un medio seguro que permita de una forma rápida y fiable la aplicación de los exámenes parciales de programación u otra materia. La realización de dichos exámenes se efectúa de una forma poco segura y en algunos casos puede llegar a ser ineficiente, trayendo como consecuencia que no se explote al máximo el tiempo con el que cuenta la realización de cualquier examen de alguna de las disciplinas docentes que se imparten.

A partir de la situación problemática antes expuesta se formula el siguiente **Problema a resolver**: ¿Cómo planificar las evaluaciones prácticas docentes convocadas por el Departamento de Programación de la Facultad 4 de la Universidad de las Ciencias Informáticas?

El **Objeto de Estudio** del presente trabajo se enfocará en:"Los procesos de evaluación docente convocados por el Departamento de Programación de la Universidad", teniendo como **Campo de Acción**:"La aplicación de exámenes prácticos convocadas por el Departamento de Programación de la Facultad 4 de la Universidad de las Ciencias Informáticas".

Para responder al problema anteriormente planteado, los autores se trazaron como **Objetivo General** "Realizar el análisis y diseño de una aplicación Web que permita planificar y controlar los procesos de evaluación docente convocados por el Departamento de Programación de la Facultad 4.

Para darle cumplimento a dicho objetivo se trazaron los siguientes **Objetivos Específicos**:

- Realizar el análisis de los requisitos funcionales del sistema informático a desarrollar.
- Diseñar el sistema analizado.

Para darle cumplimiento a lo antes descrito se han propuesto las siguientes **Tareas de la Investigación:**

- Realizar el diseño teórico de la investigación
- Identificar los requerimientos funcionales y no funcionales del sistema.
- Realizar el Diagrama de Clases del Análisis.

- Modelar y describir el Modelo de Casos de Uso del sistema.

A continuación se describen los Métodos y Técnicas investigativas utilizados en el procedimiento del presente trabajo.

Métodos Teóricos:

Analítico-Sintético: Su objetivo es analizar y resumir la situación problemática, permitiendo la extracción de los elementos más importantes

Modelación: Mediante este método se logró la creación de los modelos que permitieron visualizar, descubrir y estudiar nuevas relaciones y cualidades de la aplicación en cuestión.

Métodos Empíricos:

Entrevistas a Expertos: Se sostuvo una conversación planificada con las personas más adiestradas al tema (lideres de proyectos) ya que son las más conocedoras acerca de los requerimientos necesarios.

Observación: Se realizaron una serie de observaciones a sitios relacionados con la aplicación.

Con el cumplimiento satisfactorio de los objetivos trazados se espera como **Posible Resultado.** Obtener el análisis y diseño de una herramienta para planificar y controlar la ejecución de las evaluaciones prácticas docentes

Para lograr la comprensión y claridad de los contenidos de la investigación realizada se ha estructurado el documento en tres capítulos. En el Capítulo 1: Fundamentación teórica, se argumenta sobre las herramientas, metodologías y tecnologías a utilizar. En el Capítulo 2: Descripción de la Solución se describe el sistema propuesto para apoyar la planificación y control de la aplicación de los exámenes prácticos que realiza la facultad en los laboratorios docentes, se argumentan algunas de las tantas terminologías que se utilizan y se ejecuta la descripción de los casos de uso.En el Capítulo 3: Diseño del Sistema Propuesto se define cómo el sistema cumplirá con los requisitos establecidos, para así satisfacer las necesidades del cliente. El mismo está constituido por el diseño, el cual se encarga de definir cómo interactúan las clases y por la arquitectura que llevará a cabo dicho sistema. Además se abordan los patrones a utilizar y una explicación de la arquitectura propuesta

Para cada capítulo se ofrecen sus conclusiones parciales y al final del documento se exponen las conclusiones generales, las recomendaciones y el glosario de términos.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

Introducción

En el presente capítulo se realiza un estudio del estado del arte de acuerdo al objeto de estudio y campo de acción, haciendo referencia a las tendencias y tecnologías que se emplean actualmente y han sido propuestas en el presente trabajo para que puedan formar parte de la propuesta de solución del problema científico. Además se analizan los principales servidores Web y las herramientas que se necesitan para desarrollar el sistema informático requerido, de esta manera se seleccionan las técnicas, metodología y herramientas que serán empleados para cumplir los objetivos propuestos.

1.2 Entornos virtuales de aprendizaje.

Un Ambiente Virtual de Aprendizaje (AVA) ó *Virtual learning environment* (VLE) es un sistema de software diseñado para facilitar a profesores la gestión de cursos virtuales para sus estudiantes, especialmente en la administración y desarrollo del curso. Los Ambiente Virtual de Aprendizaje pueden seguir a menudo el progreso de los principiantes, pueden ser controlados por los profesores y los mismos estudiantes. Originalmente diseñados para el desarrollo de cursos a distancia, vienen siendo utilizados como suplementos para cursos presenciales.

Estos sistemas funcionan generalmente en el servidor, para facilitar el acceso de los estudiantes a través de Internet.

Estos Ambientes Virtuales, se basan en el principio de aprendizaje colaborativo donde se permite a los estudiantes realizar sus aportes y expresar sus inquietudes en los foros, además van apoyados de herramientas multimodales que hagan más agradable el aprendizaje pasando de ser simplemente un texto en línea, a un entorno interactivo de construcción de conocimiento. (Morgan, 2003). Sistemas automatizados existentes.

Para la realización de la presente investigación se realizó un estudio relacionado con los Ambiente Virtual de Aprendizaje existentes a nivel internacional nacional así como a nivel de universidad.

1.2.1 Sistemas existentes a nivel internacional.

Claroline es un groupware asíncrono y colaborativo. Proyecto de software libre que se distribuye con licencia GNU/GPL. Está escrito en el lenguaje de programación PHP, utiliza MySQL como SGBD. Sigue las especificaciones de SCORM e IMS. Está disponible para plataformas (Linux) y navegadores libres (Mozilla, Netscape), y plataformas (Unix, Mac OS X y Windows) y navegadores propietarios (Internet Explorer (Marcel Lebrun , 2005)

Presenta las características propias de un sistema de gestión de contenidos (CMS). Puede ser utilizado por formadores, para administrar cursos virtuales en entornos e-learning que permite:

- Publicar documentos en cualquier formato: word, pdf, html, vídeo, etc.
- Administrar foros de discusión tanto públicos como privados.
- Administrar listas de enlaces.
- Crear grupos de estudiantes.
- Confeccionar ejercicios.
- Estructurar una agenda con tareas y plazos.
- Hacer anuncios, vía correo electrónico por ejemplo.
- Gestionar los envíos de los estudiantes: documentos, tareas, trabajos, etc.
- Crear y guardar chats.

Moodle es una aplicación que pertenece al grupo de los Gestores de Contenidos Educativos (LMS, Learning Management Systems), también conocidos como Entornos de Aprendizaje Virtuales (VLE, Virtual Learning Managements), un subgrupo de los Gestores de Contenidos (CMS, Content Management Systems).

De forma general , se puede decir que Moodle es una aplicación para crear y gestionar plataformas educativas, es decir, espacios donde un centro educativo, institución o empresa, gestiona recursos educativos proporcionados por unos docentes y organiza el acceso a esos recursos por los estudiantes, y además permite la comunicación entre todos los implicados (alumnado y profesorado).

Características de Moodle.

Entorno de aprendizaje modular y dinámico orientado a objetos, sencillo de mantener y actualizar.

Excepto el proceso de instalación, no necesita prácticamente de "mantenimiento" por parte del administrador.

Dispone de una interfaz que permite crear y gestionar cursos fácilmente.

- Los recursos creados en los cursos se pueden reutilizar.
- La inscripción y autenticación de los estudiantes es sencilla y segura.
- Resulta muy fácil trabajar con él, tanto para el profesorado como el alumnado.

Detrás de este entorno de aprendizaje hay una gran comunidad que lo mejora, documenta y apoya en la resolución de problemas.

Está basado en los principios pedagógicos constructivistas: el aprendizaje es especialmente efectivo cuando se realiza compartiéndolo con otros. (Moodle, 2006)

1.2.2 Sistemas existentes a nivel nacional.

En la actualidad Cuba está inmersa en el inmenso mundo de la informática. El desarrollo y producción de software en este país se encuentra en una fase de madurez y aún tiene mucho camino por recorrer, no obstante ya se observa la aplicación de medios informáticos para contribuir a la adecuada formación de los estudiantes cubanos con nuevos modelos presenciales, ejemplo de ellos son los que a continuación se describen:

UVS Cubana: La Universidad Virtual de Salud es una institución académica virtual, que cuenta con la participación activa y creadora de las instituciones académicas, docente-asistenciales e investigativas del Sistema Nacional de Salud Cubano, para desarrollar sistemas de programas de Educación en Red, con el empleo de las Tecnologías de Información y Comunicación, que posibilitan la educación posgraduada de forma masiva; así como interconsultas y discusiones clínicas con fines docentes y el empleo de métodos activos y participativos de aprendizaje y una organización tutorial de apoyo. (Salud)

Grupo de Tecnología Educativa: Tiene como objetivo central explorar, indagar, valorar y utilizar las tecnologías de la información y la comunicación para apoyar o desarrollar programas a distancia, tanto de pregrado como de postgrado. (Educativa)

1.2.3 Sistemas existentes a nivel de universidad.

EVA: Éste es un espacio de apoyo al proceso de formación de la Carrera de Ingeniería en Ciencias Informáticas (UCI), donde los profesores pueden implementar estrategias de enseñanza-aprendizaje complementarias a las clases presenciales, así como diseñar cursos semipresenciales o totalmente a distancia, disponiendo los estudiantes de un poderoso medio en el cual pueden obtener, utilizar o compartir materiales (EVA)

Análisis sobre la situación de la planificación y control de los procesos de evaluación docentes convocados por el departamento de Programación en la Universidad de las Ciencias Informáticas.

Hoy Cuba está inmersa en el inmenso mundo de la informática. El desarrollo y producción de software en Cuba se encuentra en una fase de madurez y aún tiene mucho camino por recorrer. En la Universidad de las Ciencias Informáticas, específicamente en la Facultad 4, no se cuenta con una herramienta que facilite de una manera rápida, eficiente y segura la aplicación de exámenes parciales de programación, lo que constituye una de las dificultades y uno de los peldaños fundamentales para llevar los procesos evaluativos docentes a la mayor eficiencia y madurez posible.

1.3 Metodologías de Desarrollo de Software

Las metodologías se desarrollan con el objetivo de dar solución a los problemas existentes en la producción de software, que cada vez son más complejos. Estas engloban procedimientos, técnicas, documentación y herramientas que se utilizan en la creación de un producto de software. Una metodología es un proceso, y un proceso de desarrollo de software es la definición del conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, a modo de plantilla que explica los pasos necesarios para terminar el proyecto

1.3.1 Fundamentación de la metodología a utilizar.

Para construir un sistema con la calidad requerida, que cumpla todas las expectativas de los usuarios a los que va dirigido, se impone la necesidad de definir la metodología de desarrollo de software que guiará el proceso de automatización. Para guiar la propuesta que presenta este trabajo, se realizó un estudio de las diferentes metodologías de desarrollo de software conocidas mundialmente. Dentro de las misma se pueden citar: RUP (Proceso Unificado de Desarrollo de Software) y XP (eXtreme Programming ó Programación Extrema).

1.3.2 Programación Extrema (XP, eXtreme Programming).

XP es una de las metodologías ligeras [1]de desarrollo de software utilizadas para proyectos de corto plazo y corto equipo. La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. Intenta reducir la complejidad del software por medio de un trabajo orientado directamente al objetivo, basado en las relaciones interpersonales y la velocidad de reacción. (ESCRIBANO, 2002)

Es el más destacado de los procesos ágiles de desarrollo de software. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. XP es recomendada para aumentar la velocidad de desarrollo de un producto, el cual fundamenta su desarrollo en los casos de prueba y en las historias de usuario (ANDRES., 2004).

Ciclo de vida de un proyecto XP

El ciclo de vida ideal de XP consiste de seis fases:

- ✓ Exploración: En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.
- ✓ Planificación de la Entrega (*Release*):En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente.
- ✓ Iteraciones:Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores.

[1] Metodología Ligera: Se entiende como desarrollo ágil de software a un paradigma de desarrollo de software basado en los procesos ágiles.

- ✓ Producción:La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase. Es posible que se rebaje el tiempo que toma cada iteración, de tres a una semana. Las ideas que han sido propuestas y las sugerencias son documentadas para su posterior implementación .
- ✓ Mantenimiento:Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura.
- ✓ Muerte del Proyecto**:**Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo.

Ventajas:

- Programación organizada.
- Menor taza de errores.
- Satisfacción del programador.

Desventajas:

- Es recomendable emplearlo solo en proyectos a corto plazo.
- Altas comisiones en caso de fallar.
- Xp tiene muchas críticas especialmente contra la programación por parejas por parte de muchos programadores con gran sentimiento de posesión del código, piensan que ellos son los mejores conocedores de las herramientas y lenguajes que utilizan y que si alguien no lo entiende es por que no sabe lo suficiente.
- Xp es mas una filosofía de trabajo que una metodología. Por otro lado ninguna de las practicas defendidas por Xp son invención de este método, Xp lo que hace es ponerlas todas juntas.

1.3.3 Proceso Unificado de Desarrollo de Software (RUP).

El Proceso Unificado de Desarrollo es una metodología del Negociador Internacional de Computadoras IBM (International Business Machines por sus siglas en inglés) para el desarrollo y construcción de software basado íntegramente en UML (Unified Modeling Language) como soporte a la metodología. En RUP un proceso define quién está haciendo qué, cuándo y cómo alcanzar un determinado objetivo. (JACOBSON, et al., 2000)

El Proceso Unificado es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto. (JACOBSON, et al., 2000)

RUP establece las actividades y los criterios para conducir un sistema desde su máximo nivel de abstracción (la idea del cliente), hasta su nivel más concreto (un programa ejecutándose). Además soporta las técnicas orientadas a objetos. RUP divide en 4 fases el desarrollo del software:

Fase de Inicio, Fase de Elaboración, Fase de Construcción, Fase de Transición:

Fase del RUP	Hito
Inicio	Objetivos
Elaboración	Arquitectura
Construcción	Funcionalidad operativa
Transición	Release1 del sistema

Tabla 1 Fases de RUP

En RUP se han agrupado las actividades en grupos lógicos definiéndose nueve flujos de trabajo principales. Los seis primeros son conocidos como flujos de ingeniería y los tres últimos como de apoyo o soporte

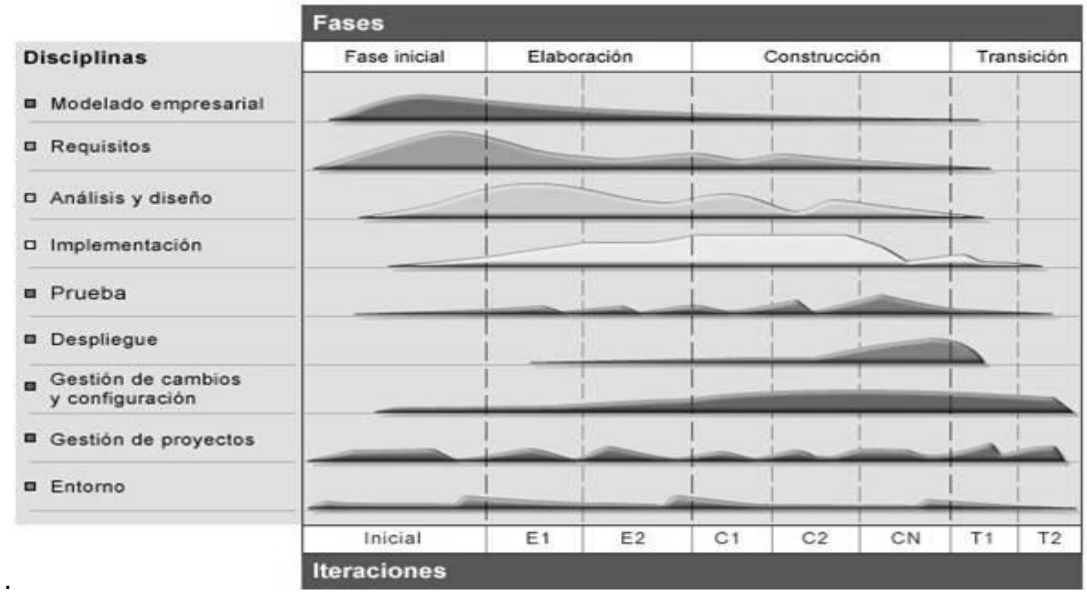


Figura 1 Fases y flujos de RUP (JACOBSON, et al., 2000)

Los flujos de trabajo de RUP son:

- ✓ Modelamiento del negocio: describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
- ✓ Requerimientos: define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
- ✓ Análisis y diseño: describe cómo el sistema será realizado a partir de la funcionalidad prevista y las restricciones impuestas (requerimientos), por lo que indica con precisión lo que se debe programar.
- ✓ Implementación: define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación.
- ✓ Prueba (Testeo): Busca los defectos a lo largo del ciclo de vida.
- ✓ Despliegue: produce release del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.
- ✓ Administración del proyecto: involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.

- ✓ Administración de configuración y cambios: describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a utilización/actualización concurrente de elementos, control de versiones, etc.
- ✓ Ambiente: contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización. (JACOBSON, et al., 2000)

Metodologías	RUP	XP
Aspecto de comparación	Nivel de funcionalidad	Nivel de funcionalidad
Descripción	Dirigido por los Casos de Uso: los requerimientos funcionales son expresados en la forma de Casos de Uso, que guían la realización de una arquitectura ejecutable de la aplicación. Centrado en la Arquitectura: el proceso focaliza el esfuerzo del equipo en construir los elementos críticos estructuralmente y del comportamiento (llamados Elementos Arquitecturales) antes de construir elementos menos importantes. Proceso Iterativo e Incremental: particiona el ciclo de vida en iteraciones que producen versiones increméntales de los ejecutables de	Re fabricación: se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio. Programación en pares: propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras. Pruebas unitarias continuas, frecuentemente repetidas y automatizadas. Programación en parejas: se recomienda

	la aplicación. Forma disciplinada de asignar tareas y responsabilidades. Desarrollo basado en componentes. Utilización de un único lenguaje de modelación. Proceso Integrado Único. Divide el proceso de desarrollo en ciclos. (JACOBSON, et al., 2000)	que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. La mayor calidad del código escrito de esta manera -el código es revisado y discutido mientras se escribe- es más importante que la posible pérdida de productividad inmediata. Frecuente integración del equipo de programación con el cliente o usuario. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo. Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes. Simplicidad en el código: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo. (ANDRES., 2004).

Tabla 2 Comparación de RUP y XP

1.3.4 Conclusiones acerca de las metodologías de desarrollo de software estudiadas.

En este caso particular se decidió la utilización de la metodología RUP por ser la metodología de mayor competencia profesional, además de que la misma se encuentra estandarizada en la universidad y dada la posibilidad de adaptabilidad brinda una disciplina para el desarrollo de todos y cada uno de los objetivos a cumplir de un proyecto. Es una guía en el proceso de desarrollo de software que posibilita el desarrollo de software a gran escala, mediante un proceso continuo de pruebas y retroalimentación, garantizando el cumplimiento de ciertos estándares de calidad. Este proceso de desarrollo constituye un marco metodológico que define en términos de metas estratégicas, objetivos, actividades y artefactos (documentación) requerido en cada fase de desarrollo permitiendo enfocar el esfuerzo de los recursos humanos en términos de habilidades, competencias y capacidades a asumir roles específicos con responsabilidades bien definidas a diferencia de XP que presenta un diseño evolutivo el cual no le da importancia al análisis como fase independiente, puesto que se trabaja exclusivamente en función de las necesidades del momento. RUP realiza un levantamiento exhaustivo de requerimientos, busca detectar defectos en las fases iniciales, intenta reducir al número de cambios tanto como sea posible, realiza el Análisis y Diseño, tan completo como sea posible. Proporciona muchas ventajas sobre XP dándole énfasis en los requisitos y el diseño. La ventaja principal de RUP es que se basa todo en las mejores prácticas que se han intentado y se han probado en el campo (en comparación con XP que se basa en las prácticas inestables que se han utilizado juntas). Además de lo planteado, se decidió la utilización de la metodología RUP por poseer un robusto y consistente nivel de funcionalidad y adaptabilidad además porque la Facultad cuenta con un equipo de desarrollo que posee una sólida experiencia en esta metodología.

1.4 Fundamentación de las herramientas a utilizar para el desarrollo del sistema.

La demanda del desarrollo de software llevó a la introducción de nuevos conceptos de interacción hombre-máquina, a la multiprogramación y a los sistemas multiusuario. La evolución de los sistemas de computadoras se caracterizó por el procesamiento distribuido (múltiples computadoras, cada una ejecutando funciones concurrentemente y comunicándose con alguna otra), incrementando notablemente la complejidad de los sistemas informáticos. Además de los diversos problemas de software que confrontaban las organizaciones en el momento, por el poco alcance para solucionar dificultades como: la cantidad de software acumulado sin desarrollar, el excesivo mantenimiento de sistemas, la falta de documentación y el envejecimiento de las aplicaciones que enfrentaban muchas empresas. Como un nuevo intento por resolver estos problemas surge una herramienta llamada CASE2 es una sigla que

corresponde a las iniciales de Computer Aided Software Engineering; y en su traducción al español significa Ingeniería de Software Asistida por Computadora.

¿Qué es CASE? Las herramientas CASE son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. CASE es también definido como el conjunto de métodos, utilidades y técnicas que facilitan el mejoramiento del ciclo de vida del desarrollo de sistemas de información, completamente o en alguna de sus fases (M.PANIAGUA, 2005)

Dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software (Investigación Preliminar, Análisis, Diseño, Implementación e Instalación). Proporciona al ingeniero la posibilidad de automatizar actividades manuales y de mejorar su visión general de la ingeniería, las herramientas CASE ayudan a garantizar que la calidad se diseñe antes de llegar a construir el producto. (PRESSMAN, 2005.)

Para el desarrollo del sistema se analizarán algunas de las herramientas CASE posibles a utilizar en su construcción, justificándose la selección de la misma, teniendo en cuenta los conocimiento, experiencia y habilidades en procesamiento de transacciones que tiene el personal del equipo y la estructura que se desea adoptar en el proyecto.

1.4.1 Lenguaje de modelado

Un lenguaje de modelado de objetos se puede definir como el conjunto estandarizado de símbolos y de modos de disponerlos para modelar (parte de) un diseño de software orientado a objetos.

El Lenguaje Unificado de Modelado es un lenguaje gráfico para visualizar, especificar, construir, documentar y comunicar los artefactos de un sistema de software (JACOBSON, et al., 2000). Utilizado para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir con la finalidad de describir modelos del sistema (del mundo real y del mundo del software), basados en los conceptos de objetos.

1.4.1.1 Lenguaje Unificado de Modelado 2.0 (UML, Unified Modeling Language).

UML es un lenguaje, que permite modelar analizar y diseñar sistemas orientados a objetos. Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Ofrece un estándar para describir un panorama del sistema modelo, incluyendo aspectos concretos como expresiones de

lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables y aspectos conceptuales como los procesos de negocios y funciones del sistema. UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software. El UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. La finalidad de los diagramas es presentar diversas perspectivas de un sistema, a las cuales se les conoce como modelo. Debido a que el UML es un lenguaje, cuenta con reglas para combinar tales elementos. Es importante recalcar que UML no es una guía para realizar el análisis y diseño orientado a objetos, es decir, no es un proceso., es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos.

UML ofrece un estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. El UML específica varios diagramas, de ellos los diagramas de Casos de Usos y los diagramas de iteración, que son los artefactos concretos a partir de los cuales creamos modelos. Los diagramas UML se visualizan por medio de las vistas (proyecciones visuales del modelo). (LARMAN, 1999).

En lo que corresponde al desarrollo de programas UML, posee elementos gráficos para soportar la captura de requisitos, el análisis, el diseño, la implementación, y las pruebas, se compone de muchos elementos de esquematización que representan las diferentes partes de un sistema de software.

De forma general las principales características son:

- Lenguaje unificado para la modelación de sistemas.
- Tecnología orientada a objetos.
- El cliente participa en todas las etapas del proyecto.
- Corrección de errores viables en todas las etapas.
- Aplicable para tratar asuntos de escala inherentes a sistemas complejos de misión crítica, tiempo real y cliente/servidor.
- Facilita a los integrantes de un equipo multidisciplinario participar e intercomunicarse fácilmente. (LARMAN, 1999).

1.4.2 Visual Paradigm for UML 6.1 Enterprise Edition.

Visual Paradigm for UML 6.1 Enterprise Edition, es un software privativo para modelado en UML 2.0. Esta herramienta tiene unas características gráficas muy cómodas, que facilitan la realización de los diagramas de modelado que sigue el estándar de UML, los mismo son: Diagramas de clase, Casos de Uso, Comunicación, Secuencia, Estado, Actividad, Componentes, etc. Se integra con las siguientes herramientas Java: (PARADIGM, 2009)

Eclipse/IBM WebSphere.

- JBuilder.
- NetBeans IDE.
- Oracle JDeveloper.
- BEA Weblogic.

1.4.3 ArgoUML.

ArgoUML es una herramienta utilizada en el modelado de sistemas, mediante la cual se realizan diseños en UML llevados a cabo en el análisis y pre-diseño de sistemas de software; incluye el soporte para todos diagramas de UML 1.4 usuales. Funciona en cualquier plataforma de Java y está disponible en diez lenguas.

1.4.4 Rational Rose.

Rational Rose es la herramienta Case desarrollada por los creadores de UML que cubren todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes y certificación de las distintas fases (SOFTWARE, 2007). Nos permite una trazabilidad real entre modelo (análisis y diseño) y el código ejecutable, domina el mercado de herramientas para el modelado, análisis, diseño y construcción orientada a objetos, tiene todas las características que los desarrolladores, analistas, y arquitectos exigen, desarrollo basado en componentes con soporte para arquitecturas líderes en la industria y modelos de componentes, facilidad de uso e integración optimizada. Es una herramienta con plataforma independiente que ayuda a la comunicación entre los miembros del equipo, a monitorear el tiempo de desarrollo y a entender el entorno de los sistemas. Una de las grandes ventajas de Rational Rose es que utiliza la notación estándar en la arquitectura de software (UML), la cual permite a los arquitectos de software y desarrolladores visualizar el sistema completo utilizando un lenguaje común,

además los diseñadores pueden modelar sus componentes e interfaces en forma individual y luego unirlos con otros componentes del proyecto.

Además propone la utilización de cuatro tipos de modelo para realizar un diseño del sistema, utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico (vista de Casos de Uso, vista Lógica, vista de Componentes y vista de Despliegue). A continuación se muestra una comparación entre las 3 herramientas CASE caracterizadas anteriormente, lo que ayudará a la fundamentación de la seleccionada para el modelado del sistema:

ArgoUML.	Rational Rose.	Visual Paradigm for UML 6.1 Enterprise Edition.
Funcionalidad Multiplataforma (*Java 1.2*) Meta modelo de UML estándar Soporte para bases de datos Exporta los diagramas a distintos formatos Generación de código (parcial) Soporte cognitivo: Proactivo (criticas de diseño, listas de cosas por hacer, correcciones automáticas) Comprensión y solución del problema (perspectivas navegacionales, vistas superpuestas, representaciones alternativas de diseño: gráficos, texto, tablas	Funcionalidad Admite como notaciones: UML, COM, OMT y Booch. Realiza Chequeo semántico de los modelos. Permite generar código a partir de modelos y viceversa. Desarrollo multiusuario. Integración con modelado de datos. Generación de documentación. Tiene un lenguaje de script para poder ampliar su funcionalidad. Disponible en múltiples plataformas	Funcionalidad Entorno de creación de diagramas para UML 2.0 Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación. Capacidades de ingeniería directa e inversa. Modelo y código que permanece sincronizado en todo el ciclo de desarrollo. Disponibilidad de múltiples versiones, para cada necesidad.

		Disponibilidad en múltiples plataformas. Soporta aplicaciones Web. Las imágenes y reportes generados, no son de muy buena calidad. Varios idiomas.

Tabla 3 Comparación de las herramientas

Se utilizó Visual Paradigm for UML 6.1 ya que es una herramienta que ofrece un entorno de creación de diagramas usando las notaciones UML 2.0, con un diseño centrado en casos de uso y enfocado además al negocio, lo cual genera un software de alta calidad. Esta herramienta fue creada para el ciclo completo de desarrollo del software que lo automatiza y acelera, permitiendo la captura de requisitos, análisis, diseño e implementación. Está diseñada para usuarios interesados en sistemas de software de gran escala, apoya los estándares más recientes de la notación UML. Además una de las políticas de la facultad es migrar a software libre y esta es una herramienta multiplataforma.

1.5 Lenguaje de programación.

Un lenguaje de programación es un idioma artificial diseñado para expresar computaciones que pueden ser llevadas a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Está formado de un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Al proceso por el cual se escribe, se prueba, se depura, se compila y se mantiene el código fuente de un programa informático se le llama programación. (Lutz, 2009)

1.5.1 Java

Java fue diseñado por la compañía Sun Microsystems Inc., con el propósito de crear un lenguaje que pudiera funcionar en redes computacionales heterogéneas (redes de computadoras formadas por más de un tipo de computadora, ya sean PC, MAC's, estaciones de trabajo, etc.), y que fuera independiente de la plataforma en la que se vaya a ejecutar. Esto significa que un programa de Java puede ejecutarse en cualquier máquina o plataforma. La programación en Java, permite el desarrollo de aplicaciones bajo el esquema de Cliente Servidor, como de aplicaciones distribuidas, lo que lo hace capaz de conectar dos o más computadoras u ordenadores, ejecutando tareas simultáneamente, y de esta forma logra distribuir el trabajo a realizar. Java como lenguaje de programación ofrece diversas características como ser fácil de usar y tomar lo mejor de otros lenguajes orientados a objetos para el desarrollo de aplicaciones. Es un lenguaje distribuido, proporcionando una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets y establecer y aceptar conexiones con servidores o clientes remotos. Java es robusto ya que fue diseñado para crear software altamente fiable. Para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Siendo un lenguaje seguro presenta barreras de seguridad en el lenguaje y en el sistema de ejecución en tiempo real.

Java es compilado en la medida en que su código fuente se transforma en una especie de código máquina. Por otra parte, es interpretado, ya que se puede ejecutar directamente sobre cualquier máquina a la cual se hayan portado el intérprete y el sistema de ejecución en tiempo real. Independiente de la arquitectura de hardware: Java está diseñado para soportar aplicaciones que serán ejecutadas en los más variados entornos de red, desde Unix a Windows NT, pasando por Mac y estaciones de trabajo, sobre arquitecturas distintas y con sistemas operativos diversos.

1.5.2 PHP.

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas Web dinámicas. Es usado principalmente en interpretación del lado del servidor pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica (PHP, 2009)

P v5.0.3	**JDK 6.0**
Diseño y Funcionalidad	Diseño y Funcionalidad

Es un lenguaje multiplataforma: capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL. *Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones). *Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda. *Es libre, por lo que se presenta como una alternativa de fácil acceso para todos. *Permite las técnicas de Programación Orientada a Objetos.	*Simple. Elimina la complejidad de los lenguajes como "C" y da paso al contexto de los lenguajes modernos orientados a objetos. *Orientado a Objetos. La filosofía de programación orientada a objetos es diferente a la programación convencional. *Familiar. C o en C++, la sintaxis de Java es muy similar al de estos. *Robusto. El sistema de Java maneja la memoria de la computadora por ti. No te tienes que preocupar por apuntadores, memoria que no se esté utilizando, etc. *Seguro. El sistema de Java tiene ciertas políticas que evitan se puedan codificar virus con este lenguaje. *Portable. Como el código es interpretado, un programa compilado de Java puede ser utilizado por cualquier computadora que tenga implementado el interprete de Java. *Independiente a la arquitectura. Al compilar un programa este código es interpretado por diferentes computadoras de igual manera, solamente hay que implementar un intérprete para cada plataforma. De esa manera Java logra ser un lenguaje que no depende de una arquitectura computacional definida. *Multithreaded. Un lenguaje que puede ejecutar diferentes líneas de código al mismo tiempo. *Interpretado. Java corre en máquina virtual, por lo tanto es interpretado.

	*Dinámico. Java no requiere que compiles todas las clases de un programa para que este funcione. Si realizas una modificación a una clase Java se encarga de realizar un Dynamic Bynding o un Dynamic Loading para encontrar las clases

Tabla 4 Comparación de los lenguajes

Se utilizará como lenguaje de programación para el desarrollo de la aplicación Java JDK 6.0 por su robustez, fiabilidad y seguridad. Java es un lenguaje de desarrollo orientado a objetos, multiplataforma. Es un lenguaje interpretado y compilado, esto quiere decir que su código fuente se transforma en algo similar al código máquina. Soporta la sincronización múltiple de hilos de ejecución (*multithreading*), lo cual da la posibilidad que un hilo puede ocuparse de interactuar con el cliente y otro de realizar una operación. Otra de las tantas características que lo hacen idóneo para su utilización, es que se utiliza para dos tipos de programas, aplicaciones independientes y applets.Java es el lenguaje de programación usado en los procesos evaluativos que convoca la Facultad, por tanto tenemos una base sólida y amplio dominio del mismo.

1.6 Frameworks.

Un framework es un término muy utilizado últimamente en el campo de la informática, se utiliza para referirse a un conjunto de bibliotecas, que se utilizan para implementar la estructura de un modelo para una aplicación. Esto se realiza con el objetivo de promover la reutilización de código, posibilitando que no sea necesario perder tiempo en reinventar la rueda. Existen diferentes tipos de framework, para diferentes propósitos, algunos orientados al desarrollo de aplicaciones web, o para un determinado sistema operativo o lenguaje. En un sentido muy amplio el framework que se utiliza determina la arquitectura del software.

1.6.1 Spring Framework.

Es un framework de código abierto orientado al desarrollo de aplicaciones para la plataforma Java. Fue creado por Rod Johnson, quien lo describió por primera vez en su libro “Expert One-on-One Java EE Design and Development”. Spring; es un framework que tiene el objetivo de facilitar la construcción de aplicaciones Java, presenta un entorno diseñado para aumentar la productividad, liberando al desarrollador de tareas repetitivas, ayudándolo a hacer diseños más consistentes. Se puede utilizar en

cualquier tipo de aplicación, y es ligero por el mínimo impacto que tiene en las aplicaciones. Spring se basa en la técnica Inversión de Control (loC), técnica que promueve el bajo acoplamiento a partir de la inyección de dependencias (DI) entre los objetos y una implementación de desarrollo según el paradigma de Orientación a Aspectos (AOP) que presenta una estructura simplificada para el desarrollo y utilización de aspectos (módulos multiple object crosscutting).Es el más popular y el más ambicioso de todos los framework de peso ligero. Es el único framework que interviene en todas las capas arquitectónicas de una aplicación JEE. Además está diseñado para facilitar una flexibilidad arquitectónica (JOHNSON, 2004)

Los principales valores de Spring, según Rod Johnson, se pueden resumir en: no es agresivo, provee un modelo consistente de programación, ayuda a promover la reusabilidad de código, facilita el diseño Orientado a Objetos en aplicaciones JEE, permite la extracción de valores de configuración desde el código java a archivos XML3 o archivos de propiedades, está diseñado a fin de que las aplicaciones lo usen para que las pruebas sean lo más fácil posible.Spring hace de soluciones existentes un uso más fácil, dentro de una arquitectura consistente

1.6.2 Hibernate

Hibernate es una herramienta de Mapeo objeto-relacional para la plataforma Java, que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones. Hibérnate es software libre, distribuido bajo los términos de la licencia GNU LGPL. Hibernate busca solucionar el problema de la diferencia entre los dos modelos de datos coexistentes en una aplicación: el usado en la memoria de la computadora (orientación a objetos) y el usado en las bases de datos (modelo relacional). Hibernate está diseñado para ser flexible en cuanto al esquema de tablas utilizado, para poder adaptarse a su uso sobre una base de datos ya existente. También tiene la funcionalidad de crear la base de datos a partir de la información disponible. Hibernate es uno de estos frameworks ampliamente utilizados en el desarrollo de aplicaciones Java y aplicaciones empresariales que da solución a todos estos requerimientos típicos de una estrategia OR. (HIBERNATE, 2006.)

1.7 Sistema Gestor de Base de Datos.

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

Ventajas de su uso:

Proveen facilidades para la manipulación de grandes volúmenes de datos entre éstas:

- Simplifican la programación de equipos de consistencia.
- Manejando las políticas de respaldo adecuadas, garantizan que los cambios de la base serán siempre consistentes sin importar si hay errores correctamente, etc.
- Organizan los datos con un impacto mínimo en el código de los programas.
- Bajan drásticamente los tiempos de desarrollo y aumentan la calidad del sistema desarrollado si son bien explotados por los desarrolladores.
- Usualmente, proveen interfaces y lenguajes de consulta que simplifican la recuperación de los datos.

Conjuntos de programas que permiten crear y mantener una Base de datos, asegurando su integridad, confidencialidad y seguridad. Por tanto debe permitir:

- Definir una base de datos: especificar tipos, estructuras y restricciones de datos.
- Construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD.
- Manipular la base de datos: realizar consultas, actualizarla, generar informes.

Así se trata de un software de propósito general. Ejemplo de SGBD son Oracle y SQL Server de Microsoft, PostgreSQL.

1.7.1 PostgreSQL v8.3

Es un sistema diseñado para administrar grandes cantidades de datos, que tiene la fama de ser la base de datos de código abierto (Open Source) más avanzada del mundo. El PostgreSQL se ejecuta en la mayoría de los Sistemas Operativos más utilizados en el mundo incluyendo Linux, varias versiones de UNIX y por supuesto Windows. Se ha preocupado por ser una solución real a los complejos problemas del mundo empresarial y a la vez mantener la eficiencia al consultar los datos. Con ese fin, se han desarrollado y añadido al PostgreSQL las más interesantes y útiles características que antes sólo podían hallarse en sistemas manejadores de bases de datos comerciales como Oracle, DB2 o Sybase, lo cual lo

coloca, como su lema indica, como "el manejador de bases de datos de código abierto más avanzado del mundo". (PostgreSQL, 2007)

1.7.2 MySQL v5.0.3

Es un sistema de gestión de bases de datos relacionales, multihilo y multiusuario, es muy utilizado en aplicaciones Web implementadas en Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación Web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. Es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones Web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

PostgreSQL v8.3	MySQL v5.0.3
Diseño y Funcionalidad	Diseño y Funcionalidad
BMS Objeto-Relacional. PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Altamente Extensible. PostgreSQL soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario. PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos. API. La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL. Estas interfaces incluyen Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C. Cliente/Servidor. PostgreSQL usa una arquitectura	Es un sistema de administración relacional de bases de datos Es software de fuente abierta Disponibilidad en gran cantidad de plataformas y sistemas. Diferentes opciones de almacenamiento según si se desea velocidad en las operaciones o el mayor número de operaciones disponibles. Conectividad segura. Replicación. Búsqueda e indexación de campos de texto. Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente. Seguridad: ofrece un sistema de contraseñas y privilegios seguro mediante verificación basada en el host y el tráfico de contraseñas está cifrado al conectarse a un servidor (MySQL, 2009)

proceso-por-usuario cliente/servidor.	

Tabla 5 Comparación de los SGBD

Se utilizará como gestor de Base Datos el PostgreSQL v8.3 por su alta extensibilidad, integridad y robustez.

Como servidor Web se seleccionó el Apache Tomcat v6.0.18. Es un servidor con soporte de servlets y JSPs. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. Tomcat puede funcionar como servidor Web por sí mismo. Es usado como servidor Web autónomo en entornos con alto nivel de tráfico y alta disponibilidad. Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java. (Apache-Tomcat, 2009)

1.8 Conclusiones del capítulo.

En este capítulo se encuentra la fundamentación de las herramientas y lenguajes que se utilizarán para solucionar el problema anteriormente planteado En los análisis realizados a sistemas similares existentes a escala nacional y de nivel UCI se concluyó que ninguno se ajusta al proceso de aplicación de exámenes prácticos de Programación de la Facultad 4, por lo que es necesario realizar un sistema que satisfaga las necesidades de la misma.

CAPITULO 2: Descripción de la solución.

2.1 Introducción

En el presente capítulo se describe el sistema propuesto para apoyar la planificación y control de la aplicación de los exámenes prácticos que realiza la facultad en los laboratorios docentes.

El sistema que se desea proponer deberá convertirse en una herramienta útil para los profesores de la especialidad, por lo tanto debe dar varías opciones de uso que no limiten su aplicación a la mera descarga y subida de ficheros. Las funcionalidades del sistema deberán abarcar la administración de las evaluaciones prácticas que se apliquen, así como la gestión de los distintos grupos de estudiantes y configuraciones de red con que desee trabajar, debe abarcar además la posibilidad de dar seguimiento sobre la marcha a un examen y de los estudiantes que en él intervienen de forma individual, así como detectar y notificar algunas acciones que se desarrollan durante la realización de una actividad de este tipo, el sistema deberá dar la posibilidad también a los estudiantes de hacer salvas de seguridad para evitar la pérdida innecesaria de su examen y permitirá también a los profesores obtener todas las soluciones hechas por los estudiantes de sus grupos.

Teniendo en cuenta la descripción anterior se definió un listado de las posibles características que se buscan en el sistema a proponer, estas características conforman los requisitos candidatos del mismo.

2.2 Requisitos candidatos del sistema

- Gestionar configuraciones de grupos de estudiantes de forma tal que estas configuraciones se puedan usar para programar más de una tarea.
- Búsqueda de grupos de estudiantes utilizando los servicios web existentes.
- Búsqueda de un estudiante por su usuario utilizando los servicios web existentes.
- Crear grupos atípicos para acceso a la realización de una tarea a varios estudiantes puntuales que no pertenezcan al mismo grupo.
- Crear una contraseña “local” a un estudiante de forma tal que le sea posible entrar con su usuario de dominio a la aplicación sin autenticar contra LDAP.

- Gestionar configuraciones de redes de forma tal de que estas configuraciones se puedan usar para programar más de una tarea.
- Las configuraciones de red pueden ser por sub-red (10.32.16.0), por rango de IP (10.32.13.1 – 10.32.13.10), o por IP estático (10.32.15.1).
- Gestionar planificaciones de exámenes.
- Especificar fecha e intervalo de horas donde el examen estará activo.
- Incorporar Materiales al examen.
- Utilizar en la misma una configuración de grupos previamente hecha.
- Utilizar en la misma una configuración de redes previamente hecha.
- Invalidar el acceso a un Estudiante determinado.
- Mostrar histórico de exámenes realizadas.
- Filtrar por fecha.
- Filtrar por estado (Planificada, En Ejecución, Ejecutada)
- Ver detalles del examen (Materiales, configuraciones de grupos y de redes).
- Monitorear las actividades en Ejecución.
- Listar las actividades en ejecución.
- Listar los grupos que pertenecen a una actividad dada.
- Por cada una de las conexiones que establezca un estudiante:
    - Mostrar IP
    - Horas de apertura y cierre.
    - Número de materiales descargados (Fichero y Hora)
    - Número de soluciones subidas (Fichero y Hora)
- Mostrar informe de asistencia a un examen.

- Graficar.
- Imprimir.
- Detectar una de las siguientes anomalías y generar una notificación. (Estas se generan durante la autenticación)
- Un estudiante al conectarse lo hace desde un IP distinto. (IP anterior, IP actual, Estudiante)
- Un usuario se conecto desde un IP que aparece asociado a otro estudiante. (IP de la conexión y Estudiantes involucrados)
- Recibir las notificaciones para poder tomar acciones concretas.
- Mostrar un listado organizado por ocurrencia cada una de las notificaciones que van siendo generadas.
- Bloquear el acceso a la aplicación del o los estudiantes que vienen relacionados con la notificación.
- Imprimir informe con las notificaciones generadas en un examen.
- Descargar las soluciones de los estudiantes de un grupo.
- Mostrar la última solución subida por cada estudiante.
- Ver el histórico de soluciones subidas por un estudiante.
- Gestionar los profesores.
- Buscar profesores utilizando los servicios web existentes.
- Mostar al estudiante la tarea a la cual tiene acceso (Siempre que esta esté activa)
- Mostrar el listado de materiales que tiene la tarea para su descarga.
- Permitir subir una o varias soluciones. (Una a la vez)
- Mostrar el listado de cada una de las soluciones que vaya subiendo el estudiante a manera de salvas de seguridad, ordenadas por la Hora en que las subió.
- Permitir descargar al estudiante cualquiera de sus soluciones previas.

- Autenticación.
    - Autenticarse como Administrador, Profesor o Estudiante. (Los dos últimos mediante el dominio)

2.3 Modelo de Dominio.

Un Modelo de Dominio es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción, en la tarea construcción del modelo de dominio, presentado como uno o más diagramas de clases y que contiene, no conceptos propios de un sistema de software sino de la propia realidad física.

Los modelos de dominio pueden utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema, ya sea de software o de otro tipo. Similares a los mapas mentales utilizados en el aprendizaje, el modelo de dominio es utilizado por el analista como un medio para comprender el sector industrial o de negocios al cual el sistema va a servir.

2.3.1 Comprensión del contexto en que se implantará el sistema.

En aras de comprender mejor el contexto en el cual se desea implantar el sistema informático, se ha decidido realizar un modelo del dominio, para ello se identificaron las principales entidades, roles y asociaciones que intervienen en el negocio.

2.3.1.1 Trabajadores del negocio

Un trabajador del negocio es una abstracción de una persona (o grupo de personas), una máquina o un sistema automatizado (software); que actúa en el negocio realizando una o varias actividades que están comprendidas dentro de un caso de uso, interactuando con otros trabajadores del negocio y manipulando entidades del negocio (JACOBSON, et al., 2000). Representa un rol y desarrollan las acciones que posibilitan que se cumpla la función de un proceso de negocio.

Profesor.

2.3.1.2 Entidades.

Las entidades del negocio no son más que contenedores de información, es decir planillas, documentos, que brindan información sólida del negocio.

Examen: Se refiere a la evaluación que se planifica y aplica a un conjunto de estudiantes y que tiene lugar en una fecha determinada.

Material: Son los documentos o anexos que conforman un examen.

Solución: Son los ficheros que suben o envían los estudiantes con las respuestas del examen.

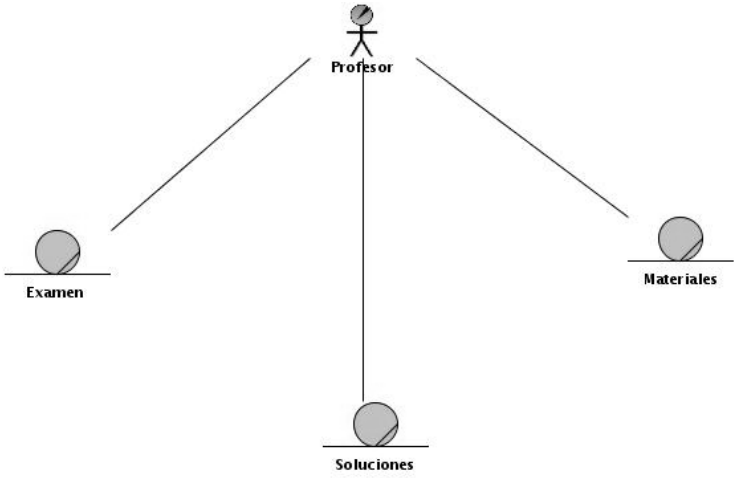


Figura 2 Diagrama de objetos del negocio.

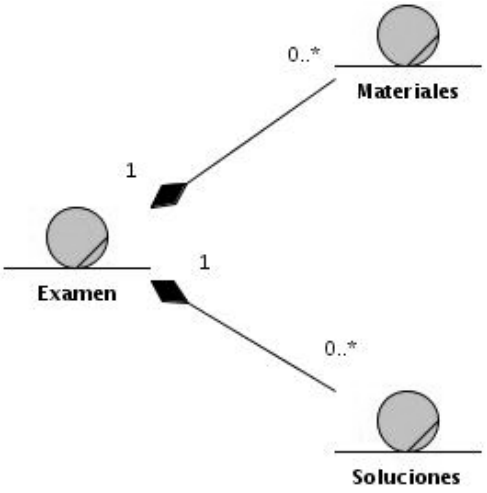


Figura 3 Diagrama de objetos del negocio con sus relaciones.

2.3.1.3. Actor del Sistema

Los actores del sistema son personas y/o otros sistemas externos que interactúan con el sistema, ven la funcionalidad del sistema y describen cómo será usado. Cada actor define un papel cohesivo y una clasificación independiente de los otros, puede participar en varios casos de uso y un caso de uso puede interactuar con varios actores**. (Profesor, Estudiante, Administrador del sistema).**

2.4. Casos de Uso del Sistema.

Los casos de uso constituyen la técnica básica y más aceptada de definición de requisitos, pues son el resultado del análisis de las necesidades de los usuarios. Permiten mostrar el contorno (actores) y el alcance (requisitos funcionales expresados como casos de uso) de un sistema. Los casos de uso describen la secuencia de interacciones que se producen entre el sistema y los actores del mismo para realizar una determinada función.

- **Administración del sistema.**
- Autenticación.
- Gestionar profesores.
- **Administración de configuraciones.**
- Gestionar configuraciones de grupos.
- Registrar o modificar configuración de grupos.
- Consultar detalles de configuración de grupos.
- Registrar grupos atípico.
- Registrar contraseña local para un estudiante.
- Gestionar configuraciones de redes.
- Registrar o modificar configuración de red.
- Consultar detalles de configuración de red.
- **Administración de exámenes.**
- Gestionar planificaciones de exámenes.

- Registrar o modificar planificación examen.
- Consultar detalles de planificación de examen.
- Invalidar el acceso a un examen de estudiante determinado.
- Consultar histórico de exámenes.
- Gestionar Soluciones.
- Consultar histórico de soluciones de un estudiante.
- **Informes y Notificaciones.**
- Monitorear los exámenes en ejecución.
- Generar informe de asistencia a un examen.
- Gestionar Notificaciones.

2.4.1 Descripción del sistema informático propuesto.

Con el objetivo de organizar el proceso de captura de requisitos funcionales se ha decidido dividir al sistema en diferentes áreas o módulos, estos agruparan las funcionalidades que giren en torno a fines similares. Los módulos identificados son:

Administración del sistema, Administración de configuraciones, Administración de exámenes, Informes y Notificaciones

Administración de configuraciones.

Contiene las funcionalidades necesarias para registrar las configuraciones de grupos y redes que se utilizarán más tarde en el módulo de Administración de exámenes. El objetivo de este módulo es garantizar una plena reutilización de estas configuraciones de modo que no sea necesario volver a crearlas una y otra vez cada vez que se necesite planificar un nuevo examen.

Gestión de configuraciones de grupos

Esta funcionalidad permitirá pre establecer nuevas configuraciones de grupos, así como modificar, eliminar y consultar detalles de configuraciones de grupos existentes. Una configuración de grupos no es más que una agrupación de varios grupos docentes que pueden ser típicos, o sea grupos comunes, y

grupos atípicos, estos últimos no son más que un conjunto de estudiantes puntuales de diferentes grupos que por determinadas razones es necesario presentar a un examen, por ejemplo los estudiantes que se presentan a coloquio son un subconjunto de los estudiantes de la facultad, en ese caso se crearía un grupo atípico con estos estudiantes. Además se dará la posibilidad de especificar una contraseña local para un estudiante y así evitar que el mismo no tenga acceso a la aplicación por tener problemas con su usuario del dominio.

Gestión de configuración de redes.

Esta funcionalidad permitirá pre establecer nuevas configuraciones de redes, así como modificar, eliminar y consultar detalles de configuraciones de redes existentes. Una configuración de redes no es más que un conjunto de reglas que definen desde donde estará visible un determinado examen, estas reglas pueden registrarse de diferentes formas: mediante la especificación de una subred (10.8.3.0), mediante la especificación de un rango de IP (10.8.3.1 – 10.8.3.5), o mediante la especificación de un IP estático (10.8.3.4). Una configuración de redes deberá poderse componer de cualquier combinación de estas reglas.

Administración de exámenes.

Este módulo tendrá todas las funcionalidades que permitan administrar los exámenes que se apliquen.

Gestión de exámenes.

Esta funcionalidad permitirá registrar los datos de un nuevo examen, así como modificar, consultar detalles y eliminar un examen. Un examen no es más que una actividad que se planifica en una fecha y un intervalo de horas determinado, además el mismo puede llevar asociado varios materiales, que no son más que ficheros que utilizará el estudiante a la hora de realizar el examen, además a un examen deberá asociarse una configuración de grupos y de redes determinada. La gestión de exámenes permitirá además bloquear el acceso de un integrante de un determinado grupo al examen en cuestión.

Histórico de exámenes.

Esta funcionalidad permitirá visualizar todos los exámenes que hayan sido aplicados, y dará la posibilidad de mostrar sus datos.

Informes y Notificaciones.

Este módulo recoge las funcionalidades que están encaminadas a dar seguimiento de los exámenes, generar informes de los mismos y generar notificaciones sobre la ocurrencia de algunas acciones no permitidas durante el examen.

Monitor de actividades en ejecución.

Dará la posibilidad de dar seguimiento a un examen durante su ejecución, del mismo se debe desglosar por cada uno de los estudiante que debieron presentarse la información concerniente a la actividad que este a realizado por cada una de las conexiones o sesiones que ha establecido con la aplicación, de forma tal que facilite cualquier análisis sobre la actividad de cualquier estudiante en un examen determinado a nivel de aplicación.

Gestor de Notificaciones.

Dará la posibilidad de generar una notificación al ocurrir algún hecho de alguna significancia durante la realización del examen, por ejemplo, cuando un estudiante se autentica o abre sesión por segunda vez, pero ahora desde una dirección IP distinta, o cuando lo hace desde una dirección IP asociada con otro estudiante durante un mismo examen. Estas notificaciones quedarían visibles al profesor de forma tal que el mismo pueda analizar la situación lo más rápido posible y tomar las medidas pertinentes.

Informes.

Dará la posibilidad de generar un informe de asistencia a nivel de actividad y desglosado por grupos en base a los estudiantes que estaban planificados a presentarse a la actividad en cuestión por cada uno de los grupos.

2.4.2 Descripción de caso de uso.

Administrar Configuraciones

- ✓ Administrar Configuraciones de grupos

Descripción del caso de uso. Gestionar configuraciones de grupos.

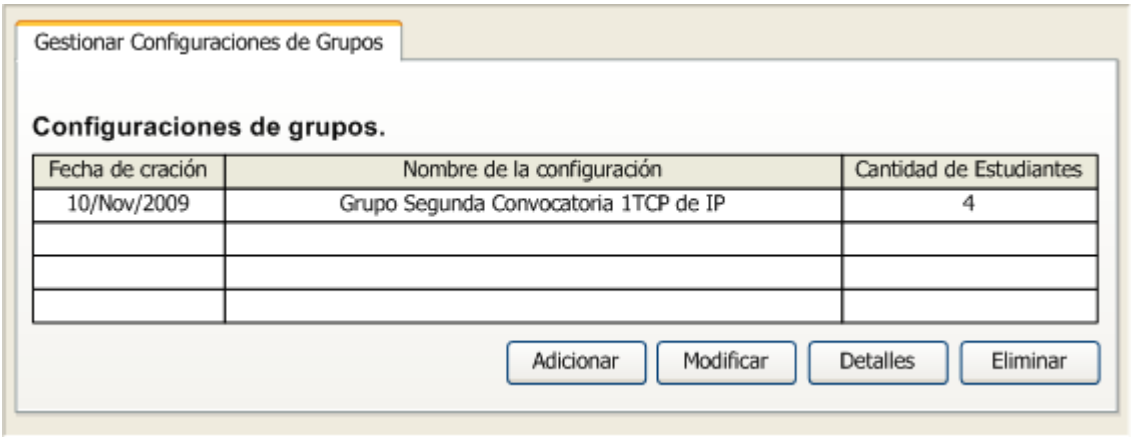
Caso de uso:	Gestionar configuraciones de grupos.

Actores:	Profesor
Resumen:	Muestra un listado con todas las configuraciones existentes. Permite además registrar nuevas configuraciones, modificar, consultar y eliminar las existentes.
Precondiciones:	El profesor se ha autenticado correctamente en el sistema.
Referencias:	
Prioridad:	Alta
Acción que inicia el caso de uso	El profesor indica que desea gestionar las configuraciones de grupos existentes.

Flujo Normal de los eventos.	
Acción del actor	Respuesta del sistema.
	El sistema muestra un listado de todas las configuraciones existentes ordenadas según la fecha en que se generaron y permite: Registrar una nueva configuración. Modificar una configuración existente. Consultar detalles una configuración. Eliminar una configuración.
El profesor selecciona una de las opciones antes descritas, y se inicia el caso de uso correspondiente: Registrar una nueva configuración inicia el	

CUS_Registar o Modificar configuración de grupos. Modificar una configuración existente inicia el *CUS_Registrar o Modificar configuraciones de grupos.* Consultar detalles de una configuración inicia el *CUS_Consultar detalles de configuración de grupo.* Eliminar una configuración, ver flujo alterno: Eliminar *Configuración.*	

Prototipo de interfaz.



Flujos Alternos.	
2.a Eliminar Configuración	
Acción del actor	Respuesta del sistema.

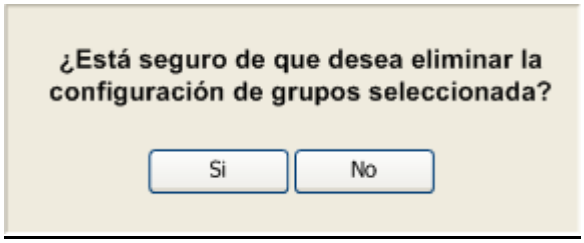
El profesor selecciona la configuración que desea eliminar e indica que desea eliminarla.	
	El sistema solicita confirmación para eliminar.
El profesor confirma la eliminación de la configuración.	
Prototipo de interfaz. 	
Pos condiciones:	Se ha registrado una nueva configuración. Se ha modificado una configuración. Se ha consultado los detalles de una configuración Se ha eliminado una configuración.

Tabla 6 Descripción del CU Gestionar configuraciones de grupos

2.5 Modelos de casos de usos

- ✓ Administración de configuraciones

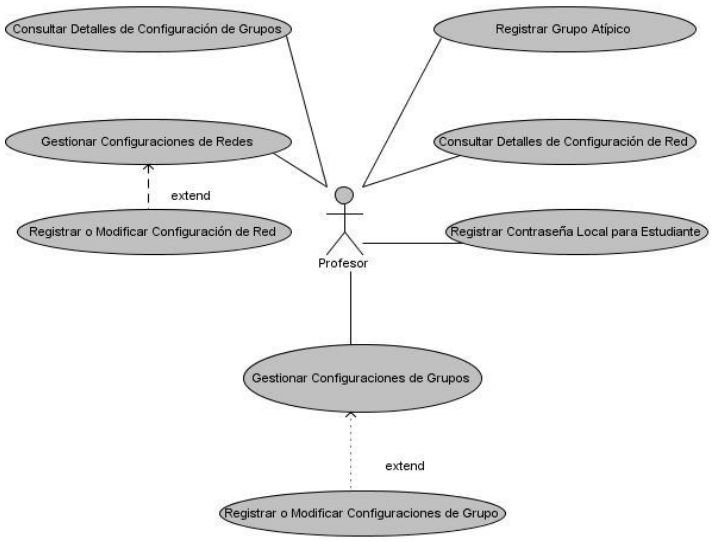


Figura 4 Modelo de CU Módulo Administración de configuraciones

- ✓ Administración de exámenes

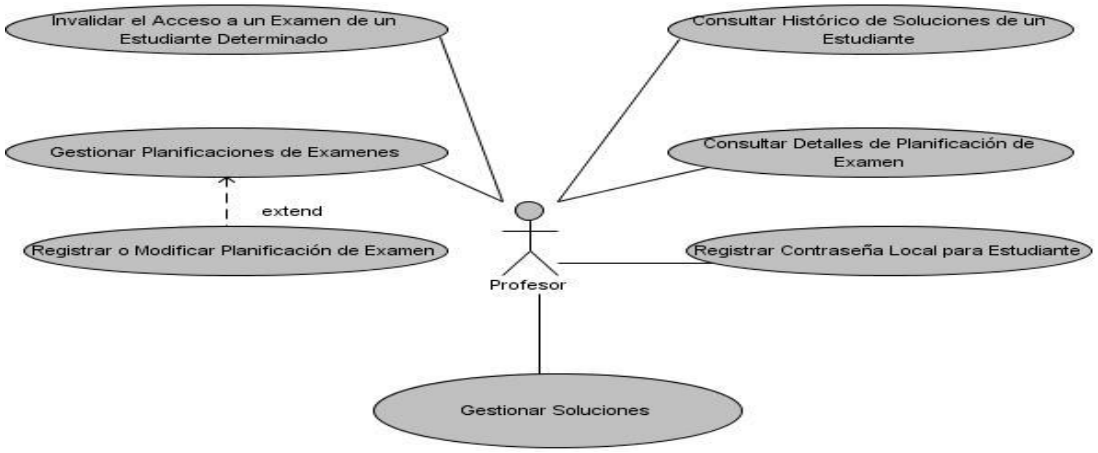


Figura 5 Modelo de CU Módulo Administración de exámenes

- ✓ Administración de sistema

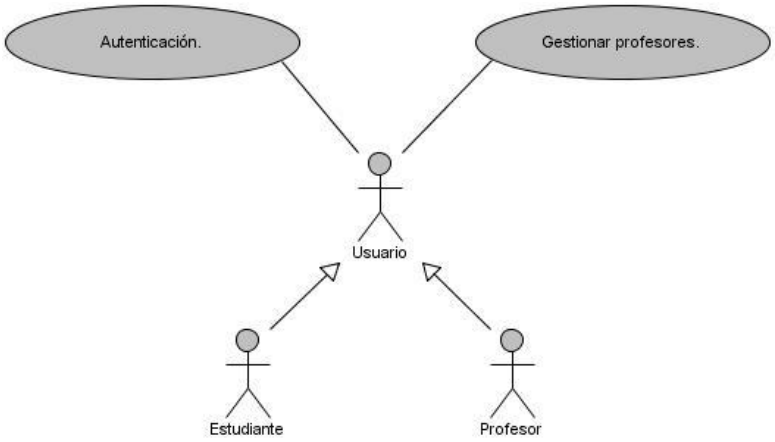


Figura 6 Modelo de CU Módulo Administración de sistemas

- ✓ Informes y Notificaciones

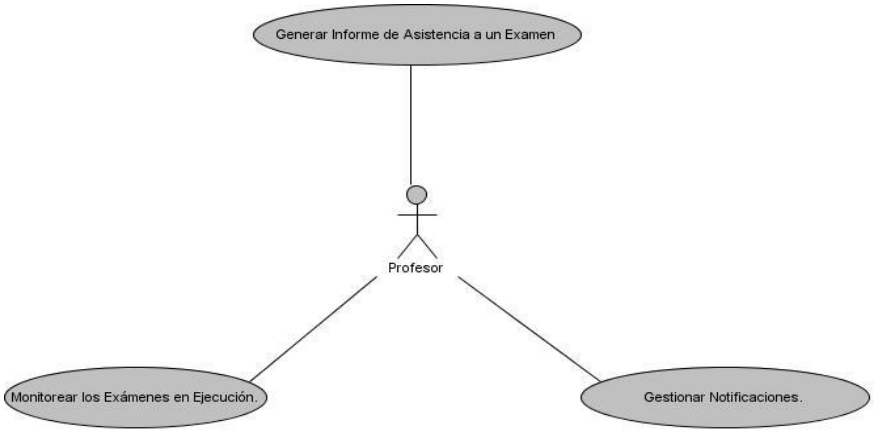


Figura 7 Modelo de CU Módulo Informe y notificaciones

2.6 Requisitos No Funcionales (RNF)

Los requerimientos no funcionales presentados a continuación constituyen propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.

RNF 1 Requerimientos de Software.

RNF 1.1 Se utilizará un Servidor Web Apache de soporte al servidor Tomcat.

RNF 1.2 Se utilizará un servidor de Base de Datos con software libre (PostgreSQL v8.3).

RNF 1.3 Para que el sistema corra es necesario computadoras donde el sistema operativo sea de la familia Windows 95, 98, NT, XP o Linux Debian o Ubuntu.

RNF 1.4 En las terminales clientes para la utilización del sistema es necesario disponer de cualquier navegador Web.

RNF 2 Requerimientos de Hardware.

RNF 2.1 Red de área local.

RNF 2.2 Máquina con procesador Pentium IV o superior para el servidor y los clientes.

RNF 2.3 Se requiere de una impresora. RNF

RNF 2.4 Memoria RAM 256 Mb en clientes y 512 Mb como mínima en servidores.

RNF 3 Requerimientos de Apariencia o interfaz externa.

RNF 3.1 Cada página no debe exceder 100 Kb en las imágenes.

RNF 3.2 Interfaz amigable, fácil de usar, sencilla, interactiva y debe mantener el mismo formato en todas las páginas.

RNF 3.3 Se utilizará java script y CSS 2.0 o superior como hoja de estilo en cascada. RNF 4 Requerimientos de Seguridad.

RNF 4.1 Para la seguridad de la información que se maneja en el sistema se definirán grupos de usuarios, los cuales tendrán asignados permisos de acción sobre cada información manejada por el sistema, para lo cual se requiere la autenticación del usuario.

RNF 4.2 El sistema debe comunicarse usando un protocolo seguro (https), durante la autenticación del usuario, para evitar que las contraseñas viajen por la red en texto plano.

RNF 4.3 Confidencialidad: Cada grupo de usuarios accederá a la información correspondiente a cada uno.

RNF 4.4 Integridad: En la base de dato se garantizará la integridad mediante checks y triggers que se ejecutarán a la hora de efectuar las operaciones de inserción, actualización y eliminación de la información.

RNF 4.5 Disponibilidad: Se mantendrá el servidor Web funcionando las 24 horas y la aplicación implementará mecanismos para recuperarse ante los posibles fallos que puedan ocurrir.

RNF 5 Requerimientos de Usabilidad.

RNF 5.1 El sistema no será concebido sólo para usuarios de la Universidad, sino también para los Clientes de los productos que estemos desarrollando y desplegando en este caso, por lo que es necesario que cuente con un diseño de interfaz de fácil uso.

RNF 5.2 Los grupos de usuarios se diferenciarán en las opciones que el sitio les posibilite.

RNF 6 Requerimientos de Restricciones en el diseño y la implementación.

RNF 6.1 El sistema se desarrollará en Java JDK v6.0 como lenguaje de programación.

RNF 6.2 La comunicación que habrá entre la base de datos y la aplicación, será por medio de módulos Java de acceso a datos. RNF

RNF 6.3 El gestor de Base de datos a utilizar será PostgreSQL.

RNF 7 Requerimientos de Soporte.

RNF 7.1 El sistema llevará incluido un video que servirá de ayuda para los usuarios.

RNF 7.2 El sistema constará de una fase de prueba con los clientes donde podrán ser encontrados posibles errores que se puedan presentar.

RNF 8 Rendimiento.

RNF 8.1 La búsqueda de cualquier documento y su visualización en pantalla debe ser como mínimo de 2 segundos y no debe exceder los 10 segundos.

RNF 8.2 La aplicación debe estar concebida para el consumo mínimo de recursos.

2.6.1 Técnicas de validación de requisitos

Los requisitos una vez definidos necesitan ser validados. La validación de requisitos tiene como misión demostrar que la definición de los requisitos define realmente el sistema que el usuario necesita o el cliente desea [23]. Es necesario asegurar que el análisis realizado y los resultados obtenidos de la etapa de definición de requisitos son correctos. Pocas son las propuestas existentes que ofrecen técnicas para la realización de la validación y muchas de ellas consisten en revisar los modelos obtenidos en la definición de requisitos con el usuario para detectar errores o inconsistencias. Aún así, existen algunas técnicas que pueden aplicarse para ello: En el presente trabajo la técnica de validación empleada es:

Prototipos: Algunas propuestas se basan en obtener de la definición de requisitos prototipos que, sin tener la totalidad de la funcionalidad del sistema, permitan al usuario hacerse una idea de la estructura de la interfaz del sistema con el usuario [26]. Esta técnica tiene el problema de que el usuario debe entender que lo que está viendo es un prototipo y no el sistema final.

2.7 Análisis

Análisis de arquitectura.

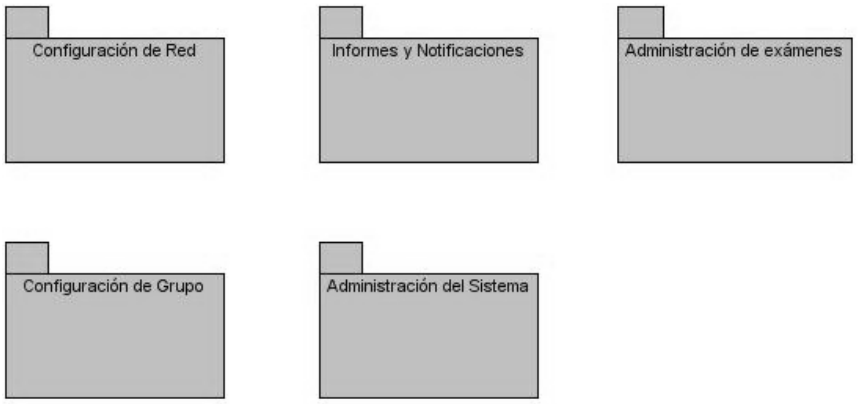


Figura 8 Diagrama de paquetes

Identificar las clases de análisis evidentes.

- Examen
- Material
- Solución

- Configuracion_Grupo
- Grupo
- Configuración Redes
- Regla
- Notificación
- Usuario

2.8 Conclusiones del capítulo.

Con el uso de RUP quedó definido: el proceso del negocio: la identificación de los actores, trabajadores y los casos de uso correspondientes. El límite de las expectativas del cliente: características que se incluyeron en el producto y el trabajo realizado para obtenerlo

La comprensión clara de las necesidades de los usuarios y los cambios pertinentes permitieron definir los requisitos funcionales y no funcionales de la aplicación que se va a desarrollar, presentando los casos de uso y sus relaciones con los actores. La validación de los requisitos, demostró que los requerimientos definidos eran los óptimos y necesarios para el desarrollo de la aplicación.

Capitulo 3 Diseño de la propuesta

3.1 Diseño de la Arquitectura

El presente capítulo se encarga de definir cómo el sistema cumplirá con los requisitos establecidos, para así satisfacer las necesidades del cliente. El mismo está constituido por el diseño, el cual se encarga de definir cómo interactúan las clases y por la arquitectura que llevará a cabo dicho sistema. El mismo juega un papel importante dentro del ciclo de desarrollo del software debido a que es este el encargado de modelar el sistema, desarrollar la arquitectura y lograr que el software cumpla con los requisitos funcionales y no funcionales además de que forma la base de la implementación.

3.2 Arquitectura del sistema.

La arquitectura definida para el desarrollo del sistema para la gestión de los proyectos productivos de la Facultad 4 es orientada al framework Spring para Java, está basada en una arquitectura de tres capas lógicas fundamentales: Capa de Presentación, Capa de Lógica de Negocio, y Capa de Acceso a Datos . Dichas capas están bien delimitadas una de la otra, una capa superior interactúa con la inferior mediante interfaces que definen las funcionalidades que la misma debe brindar.

En la arquitectura del sistema para la realización de esta Tesis se define una fachada que representa la interacción entre las Capas de Presentación y Lógica de negocio, la utilización de esta fachada cumple con el patrón de diseño Facade.

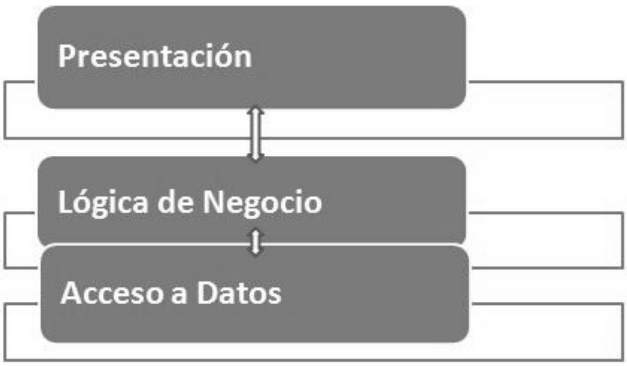


Figura 9 Arquitectura.

Capa de presentación: Esta capa es la interfaz con el usuario también se le conoce como “capa de usuario”, comunica y recolecta la información suministrada por el usuario en un mínimo de proceso (realizando validaciones para comprobar que no existen errores de formato). Esta capa se comunica únicamente con la capa de negocio llevando y trayendo los datos o registros necesarios, es la interfaz grafica del programa y debe ser lo más amena posible para una mejor comunicación con el usuario.

Lógica de negocio: específica de procesos de negocios: A veces es más oportuno para los objetos de dominio contener lógica de negocio aplicable a muchos casos de uso específicos. Sin embargo, existen funcionalidades que son realizadas en la capa de servicios de negocios. Pero en esta definición de arquitectura los objetos de dominio no presentan ningún tipo de lógica de negocio, sino que esta responsabilidad recae sobre los objetos de negocio, permitiendo usar a los objetos de dominio como objetos de transferencia que se mueven entre las capas arquitectónicas de la aplicación.

Capa de Acceso a Datos: La interacción del negocio con la capa de acceso a datos se realizará mediante el uso de interfaces. El término de Objeto de Acceso a Datos o en inglés, Data Access Object (DAO) es ampliamente usado en el desarrollo de software. Los DAO encapsulan el manejo de acceso a datos de los objetos de dominios, proveen la persistencia de los objetos transitorios y las actualizaciones de los objetos existentes en la base de datos. Las implementaciones de los DAO estarán disponibles para los objetos (típicamente para los objetos de negocio) haciendo uso de la inyección de dependencias con los objetos de negocio y las instancias de los DAO, configurada en el contenedor de inversión de control de Spring Framework.

Las interfaces de los DAO contienen básicamente los siguientes tipos de métodos:

- Métodos para descubrir: Estos localizan los objetos almacenados para ser usados por la capa de servicios de negocio.
- Métodos para persistir o salvar: Estos hacen persistentes a los objetos transitorios.
- Métodos para eliminar: Estos eliminan a los objetos guardados en el medio de almacenamiento (generalmente una base de datos).
- Métodos para conteos y otras funciones agregadas: Estos devuelven los resultados de operaciones que son más eficientes implementarlas usando funcionalidades de la base de datos que iterar sobre los mismos objetos.

3.3 Patrones.

El hombre durante su historia ha dominado muchas técnicas. El desarrollo de un software es una tarea complicada, la cual depende en gran medida de la experiencia del equipo de desarrollo. En ocasiones las operaciones se realizan de forma artesanal, los desarrolladores aprenden por un proceso de ensayo y errores y por transmisión de otros desarrolladores, se desarrollan técnicas generalmente aceptadas en el área de trabajo, lográndose un conocimiento común sobre cómo aplicar estas técnicas y además se crea una ciencia alrededor de la tarea; esto ha provocado la evolución de la utilización de estándares de solución a un problema de forma efectiva y reutilizable, es decir la utilización de un patrón que facilite la solución del problema. El patrón describe un problema que ocurre infinidad de veces en nuestro entorno, así como la solución al mismo, de tal modo que se pueda utilizar esta solución un millón de veces más adelante sin tener que volver a pensarla otra vez.

3.3.1 Patrones de casos de uso.

Los patrones de Casos de Uso son comportamientos que deben existir en el sistema, ayudan a describir qué es lo que el sistema debe hacer, es decir, describen el uso del sistema y cómo este interactúa con los usuarios. Estos patrones son utilizados generalmente como plantillas que describen cómo deberían ser estructurados y organizados los casos de uso, a continuación se describen algunos de los patrones a utilizar en el desarrollo del sistema. (LARMAN, 1999)

Reglas de negocio: Este patrón se aplica a los casos del uso que modelan los servicios que son afectados por las reglas de negocio definidas, con el propósito de capturar lo que está o no está permitido en la organización, las reglas del negocio serán definidas explícitamente de manera que será más fácil cambiar el sistema software o adaptarlo al negocio, por lo que el negocio se hace mucho más ágil. (ÖVERGAARD, 2004)

Extensión: El patrón de Extensión entre casos de uso consiste en que el comportamiento del caso de uso base se extienda bajo ciertas condiciones por otro caso de uso, especifica cómo el comportamiento definido por el caso de uso de extensión puede insertarse dentro del comportamiento definido por el caso de uso base; se utiliza entre casos de usos para ampliar los servicios que están opcionales en el sistema, permitiendo a los clientes del sistema decidir si hay que incluir un cierto servicio (ÖVERGAARD, 2004)

CRUD (Creating, Reading, Updating and Deleting): CRUD es el patrón que consiste en un caso de uso para administrar la información, modelando todas las diversas operaciones que se puedan realizar de una

parte de la información de cierta clase, tales como crearla, buscarla, modificarla y eliminarla, es utilizado cuando todos los flujos contribuyen al mismo valor de negocio y son todos cortos y simples (ÖVERGAARD, 2004)

Múltiples Actores: El patrón de Múltiples Actores cumplimenta las acciones de los actores, ellos representan como el sistema percibe su entorno Se utiliza cuando dos actores interactúan diferentemente con un caso de uso y de manera alternativa cuando los dos actores desempeñan el mismo papel hacia el caso del uso, este papel es representado por otro actor, heredado por los actores que comparten este rol (ÖVERGAARD, 2004)

3.3.2 Patrones diseño.

Los patrones de diseño proponen una forma reutilizar la experiencia de los desarrolladores, para ello clasifica y describe formas de solucionar problemas que ocurren de forma frecuente en el desarrollo. Por tanto, están basados en la recopilación del conocimiento de los expertos en desarrollo de software (LARMAN, 1999). A continuación se muestran los principales patrones de diseño utilizados en la presente solución:

Patrón Experto: Asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. (LARMAN, 1999).

Patrón Bajo Acoplamiento: Asignar las responsabilidades de forma tal que las clases se comuniquen con el menor número de clases que sea posible (LARMAN, 1999).

Patrón Alta Cohesión: Asignar a las clases responsabilidades que trabajen sobre una misma área de la aplicación y que no tengan mucha complejidad. (LARMAN, 1999).

Baja cohesión: Una clase tiene la responsabilidad exclusiva de una tarea compleja dentro de un área funcional. (LARMAN, 1999).

Alta cohesión: Una clase tiene responsabilidades moderadas en un área funcional y colabora con las otras para llevar a cabo las tareas. (LARMAN, 1999).

Patrón Controlador: Asignar la responsabilidad del manejo de mensajes de los eventos del sistema a una clase. (LARMAN, 1999).

3.4. Patrones GoF utilizados en la realización del diseño:

DAO: En software de computadores, un Data Access Object (DAO, Objeto de Acceso a Datos) es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de datos o un archivo. El término se aplica frecuentemente al Patrón de diseño Object. Los Objetos de Acceso a Datos son un Patrón de Diseño Core J2EE y considerados una buena práctica. La ventaja de usar objetos de acceso a datos es que cualquier objeto de negocio (aquel que contiene detalles específicos de operación o aplicación) no requiere conocimiento directo del destino final de la información que manipula

Patrón Estructural, Facade: Simplifica el acceso a un conjunto de clases o interfaces. Proporcionar una interfaz unificada de un subsistema sin ocultar sus interfaces. Permite acceder a elementos del sistema y realizar operaciones más complejas con ellos de forma transparente. Reduce la dependencia entre clases. Ofrece un punto de acceso al resto de clases, si éstas cambian o se sustituyen por otras, sólo hay que actualizar la clase Facade sin que el cambio afecte a las aplicaciones cliente. No oculta las clases, sino que ofrece una forma más sencilla de acceder a ellas. En los casos en que se requiere se puede acceder directamente a ellas

3.5 Modelo del diseño

El modelo del diseño es un modelo de objetos que describe la realización física de los casos de usos centrándose en como los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas en el entorno de implementación, tienen un impacto en el sistema a considerar y sirve de abstracción a la implementación y al código fuente del sistema. Contiene protocolo, cápsula, realización de casos de usos, señales, eventos, subsistema de diseño, paquetes de diseño, interfaces, clases del diseño, clases de prueba, diseño de pruebas

3.5.1 Clases del diseño

Una clase de diseño es una construcción similar a una clase en la implementación del sistema.

3.5.2 Realización de CU-Diseño.

En este tipo de diagrama de clases es en el que se modelan las relaciones que pueden existir entre las páginas Servidoras, las páginas Cliente y los Formularios. Las relaciones pueden ser de distintos tipos, entre las cuales se pueden mencionar build, submit y link. Los diagramas de clases son los más utilizados

en el modelado de sistemas orientados a objetos. Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema, esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas.

3.5.3 Extensiones de UML para Web


Server Page (página servidora): representa la página Web que tiene código, que se ejecuta en el servidor.

Client Page (página cliente): una instancia de Página Cliente es una página Web, con formato HTML.

Form (formulario): colección de elementos de entrada que son parte de una página cliente.

<<*Build*>>*:* representa una asociación especial que relaciona las páginas cliente con las páginas servidor. <<*Link*>>*:* expresa las asociaciones más comunes entre las páginas, en este caso la del hipervínculo. <<*Submit*>>*:* es la relación que se crea siempre entre una página servidor y un formulario.

La figura que se muestra a continuación representa el diagrama de diseño del modulo Configuración de Grupo del cada caso de uso del sistema Gestionar Grupos de Estudiantes.

3.6 Diagrama de Interacción.

Gestionar Grupos de Estudiantes

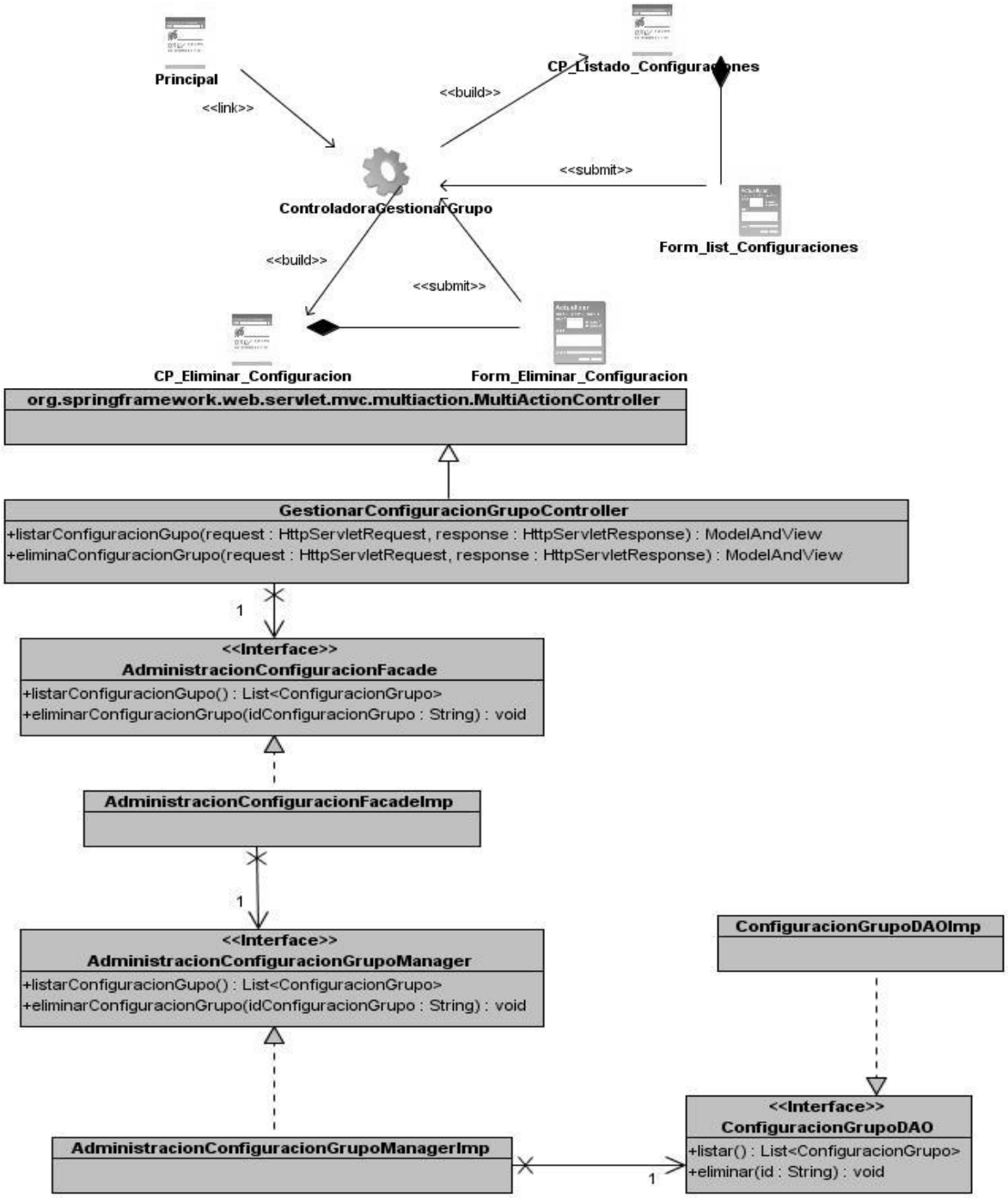


Figura 10 DI Gestionar Grupos de Estudiantes.

Los diagramas de interacción muestran cómo se comunican los objetos en una interacción. Existen dos tipos de diagramas de interacción:

- Diagramas de Colaboración: Resaltan la organización de los objetos que participan en una interacción.
- Diagramas de Secuencia: Resaltan el orden temporal de los mensajes.

Para modelar los diagramas de interacción de los casos de uso, se escoge el tipo de diagrama de secuencia debido a que los diagramas de secuencia son más adecuados para observar la perspectiva cronológica de las interacciones. La figura que se muestra a continuación representa uno de los escenarios principales de los diagramas de secuencia de los cuatros módulos del sistema para planificar y controlar la ejecución de las evaluaciones prácticas docentes.

3.6.1 Diagrama de Secuencia

Gestionar Grupos de Estudiantes

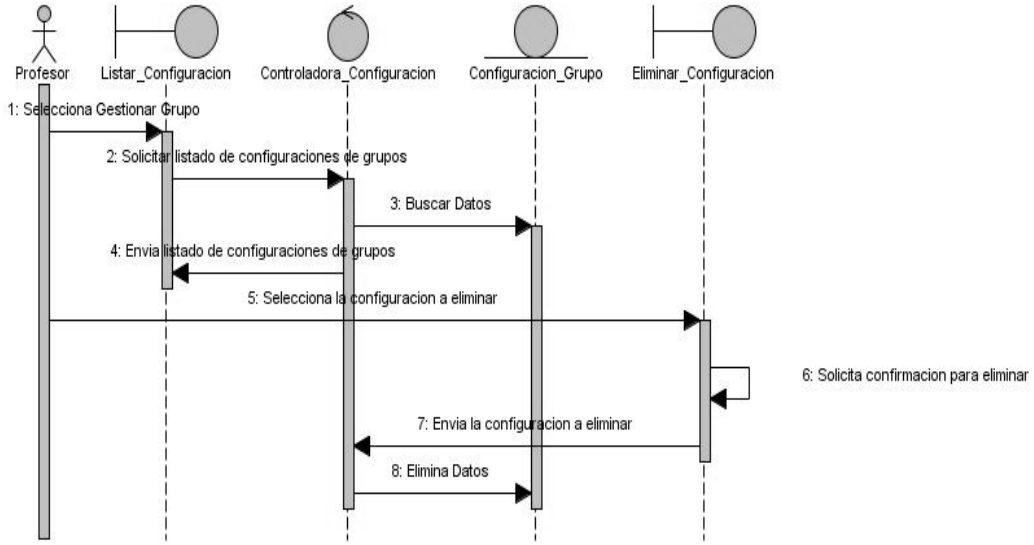


Figura 11 DS Gestionar Grupos de Estudiantes

3.7 Descripción de las tablas y sus atributos

Entidad: Examen

Descripción: Actividad que se planifica en una fecha y un intervalo de horas determinado. Un examen deberá asociarse una configuración de grupos y de redes definidas, el mismo puede llevar asociado varios materiales de consulta para el estudiante.		
Nombre	Tipo	Descripción
Fecha	Date	Este atributo representa la fecha en la que se realizara el examen
Hora de inicio	Timestamp	Este atributo representa la hora en que comenzará el examen
Hora de inicio	Timestamp	Este atributo representa la hora en que terminara el examen
Id_Configuracion_Grupo	Varchar	Este atributo es una llave foránea y es producto de la relación que existe entre esta entidad y Configuracion_Grupo.
Id_Examen	Varchar	Identificador de la tabla examen.
Id_Configuracion_Red	Varchar	Este atributo es una llave foránea y es producto de la relación que existe entre esta entidad y Configuracion_de_Red.

Tabla 7 Descripción de la tabla Examen.

Entidad: Configuracion_Grupo		
Descripción: Agrupación de varios grupos docentes que pueden ser típicos, o grupos atípicos.		
Nombre	Tipo	Descripción
Id_Configuracion_Grupo	Varchar	Identificador de la tabla Configuracion_Grupo.
Nombre	Varchar	Este atributo representa el nombre de la configuración.
Fecha_Creacion	Date	Este atributo representa la fecha de creación de la configuración.

Tabla 8 Descripción de la tabla Configuracion_Grupo.

Entidad: Grupo		
Descripción: Grupos docentes que pueden ser típicos o grupos atípicos, estos últimos no son más que un conjunto de estudiantes puntuales de diferentes grupos que por determinadas razones es necesario presentar a un examen determinado.		
Nombre	Tipo	Descripción
Id_Grupo	Varchar	Identificador de la tabla Grupo.
Nombre	Varchar	Este atributo representa el nombre o número de identificación de cada grupo.
Id_Configuracion_Grupo	Varchar	Este atributo es una llave foránea y es producto de la

		relación que existe entre esta entidad y Configuracion_Grupo.

Tabla 9 Descripción de la tabla Grupo.

Entidad: Estudiante		
Descripción :		
Nombre	Tipo	Descripción
Id_Estudiante	Varchar(20)	Identificador de la tabla Estudiante.
Nombre	Varchar(20)	Contiene el nombre de la persona
Apellidos	Varchar(20)	Apellidos de la persona
Id_Grupo	Varchar(20)	Este atributo es una llave foránea y es producto de la relación que existe entre esta entidad y Grupo.
Contrasenna_Local	Varchar	Especifica la contraseña local de un estudiante.
Nombre_Usuario	Varchar	Nombre del usuario

Tabla 10 Descripción de la tabla Estudiante.

Entidad: Material		
Descripción : Ficheros que utilizará el estudiante a la hora de realizar un examen		
Nombre	Tipo	Descripción
Nombre	Varchar	Este atributo representa el nombre que tendrá el material.
Id_Material	Varchar	Este atributo representa la llave primaria de la entidad
Fichero	Varchar	Este atributo representa el tipo de fichero.
Id_Examen	Varchar	Este atributo es llave foránea y es producto de la relación que existe entre esta entidad y Examen.

Tabla 11 Descripción de la tabla Material.

Entidad: Conexión		
Descripción: Las conexiones o sesiones que se han establecido con la aplicación cada estudiante.		
Nombre	Tipo	Descripción
Id_Conexion	Varchar	Este atributo representa la llave primaria de cada conexión.

Hora_de_Apertura	Timestamp	Este atributo representa la hora en que comenzara la conexión.
Hora_de_Cierre	Timestamp	Este atributo representa la hora en que finalizara la conexión.
Ip_Origen	Varchar	Este atributo define el origen de donde se realizara la conexión.
Id_Estudiante	Varchar	Este atributo almacena el nombre del estudiante que realizara la conexión por lo que constituye llave foránea con la entidad Estudiante
Id_Examen	Varchar	Este atributo define el examen a cual corresponde la conexión.

Tabla 12 Descripción de la tabla Conexión.

Entidad: Conexión_Material		
Descripción: Registra cada conexión por cada material consultado por un estudiante.		
Nombre	Tipo	Descripción
Id_Conexion	Varchar	Identificador de la tabla

		Conexión_Material.
Id_Material	Varchar	Este atributo indica la llave primaria de cada material que se consulte.
Hora_Conexion	Timestamp	Este atributo representa la hora en que finalizara la conexión.
Id_Estudiante	Varchar	Este atributo indica que estudiante realizara la conexión y constituye llave foránea con la entidad Estudiante.
Id_Examen	Varchar	En este atributo se almacena el id del examen, por lo que constituye la llave foránea con la entidad Examen

Tabla 13 Descripción de la tabla Conexión_Material

Entidad: Configuracion_Red		
Descripción: Un conjunto de reglas que definen desde donde estará visible un determinado examen.		
Nombre	Tipo	Descripción
Id_Configuracion_Red	Varchar	Este atributo indica la llave

		primaria de esta entidad
Nombre	Varchar	Este atributo indica nombre de cada configuración.
Fecha_Creacion	Date	Este atributo representa la fecha de creación de la configuración.
Id_Regla	Varchar	Este atributo indica el tipo de regla que definen desde donde estará visible un determinado examen.

Tabla 14 Descripción de la tabla Configuracion_Red.

Entidad: Regla		
Descripción: Estas reglas pueden registrarse de diferentes formas, mediante la especificación de una subred, de un rango de IP o la especificación de un IP estático.		
Nombre	Tipo	Descripción
Tipo	Varchar	Este atributo indica el tipo de regla.
Entidad: Regla_Ip_Statico		
Descripción: Especificación de un determinado IP.		
Nombre	Tipo	Descripción
IP_Statico	Varchar	Este atributo indica un IP

		estático
Entidad: Regla_SubRed		
Descripción: Especificación de una determinada subred.		
Nombre	Tipo	Descripción
IP_SubRed	Varchar	Este atributo indica la especificación de una subred
Entidad: Regla_Rango		
Descripción: Especificación de una determinado rango de IP.		
Nombre	Tipo	Descripción
IP_1	Varchar	Este atributo indica la especificación de un rango de IP inicial.
IP_2	Varchar	Este atributo indica la especificación de un rango de IP final.

Tabla 15 Descripción de la tabla Regla_SubRed

Entidad: Notificacion		
Descripción: Ocurrencia de algunas acciones no permitidas durante el examen. . Estas serán visibles al profesor de forma tal que el mismo pueda analizar la situación lo más rápido posible.		
Nombre	Tipo	Descripción

Tipo	Varchar	Este atributo indica el tipo de notificación.
Hora	Timestamp	Este atributo indica la hora en que se realizo la conexión.
Id_Estudiante	Varchar	En este atributo se almacena el id del estudiante, por lo que constituye la llave foránea con la entidad Estudiante
Id_Examen	Varchar	En este atributo se almacena el id del examen, por lo que constituye la llave foránea con la entidad Examen.
Id_Conexion	Varchar	En este atributo se almacena el id de la conexión, por lo que constituye la llave foránea con la entidad Conexion.

Tabla 16 Descripción de la tabla Notificacion.

Entidad: Cambio_IP
Descripción: Estudiante que se conecta desde un IP distinto, a la primera conexión.

Nombre	Tipo	Descripción
IP_Anterior	Varchar	Este atributo indica IP anterior registrado
IP_Actual	Varchar	Este atributo indica IP actual
Id_Conexion	timestamp	Este atributo indica el intervalo de tiempo en que ocurre la última conexión.

Tabla 17 Descripción de la tabla Cambio_IP.

Entidad: Usuario_Duplicado		
Descripción: Estudiante que se encuentra activo desde diferentes Ip.		
Nombre	Tipo	Descripción
IP_Sesion_Activa	Varchar	Este atributo indica IP activo.
IP_Entrante	Varchar	Este atributo indica IP entrante.

Tabla 18 Descripción de la tabla Usuario_Duplicado.

Entidad: IP_Solapado		
Descripción : Estudiante conectado desde un IP que aparece asociado a otro estudiante		
Nombre	Tipo	Descripción
IP	Varchar	Este atributo indica IP de donde

		se realizara la conexión.
Id_Estudiante2	Varchar	Este atributo indica el estudiante implicado.
Id_UltimaConexion2	Varchar	Este atributo indica la última conexión del estudiante implicado.
Id_Conexion2	timestamp	Este atributo indica el intervalo de tiempo en que ocurre la conexión.

Tabla 19 Descripción de la tabla IP_Solapado.

3.8 Modelo de datos.

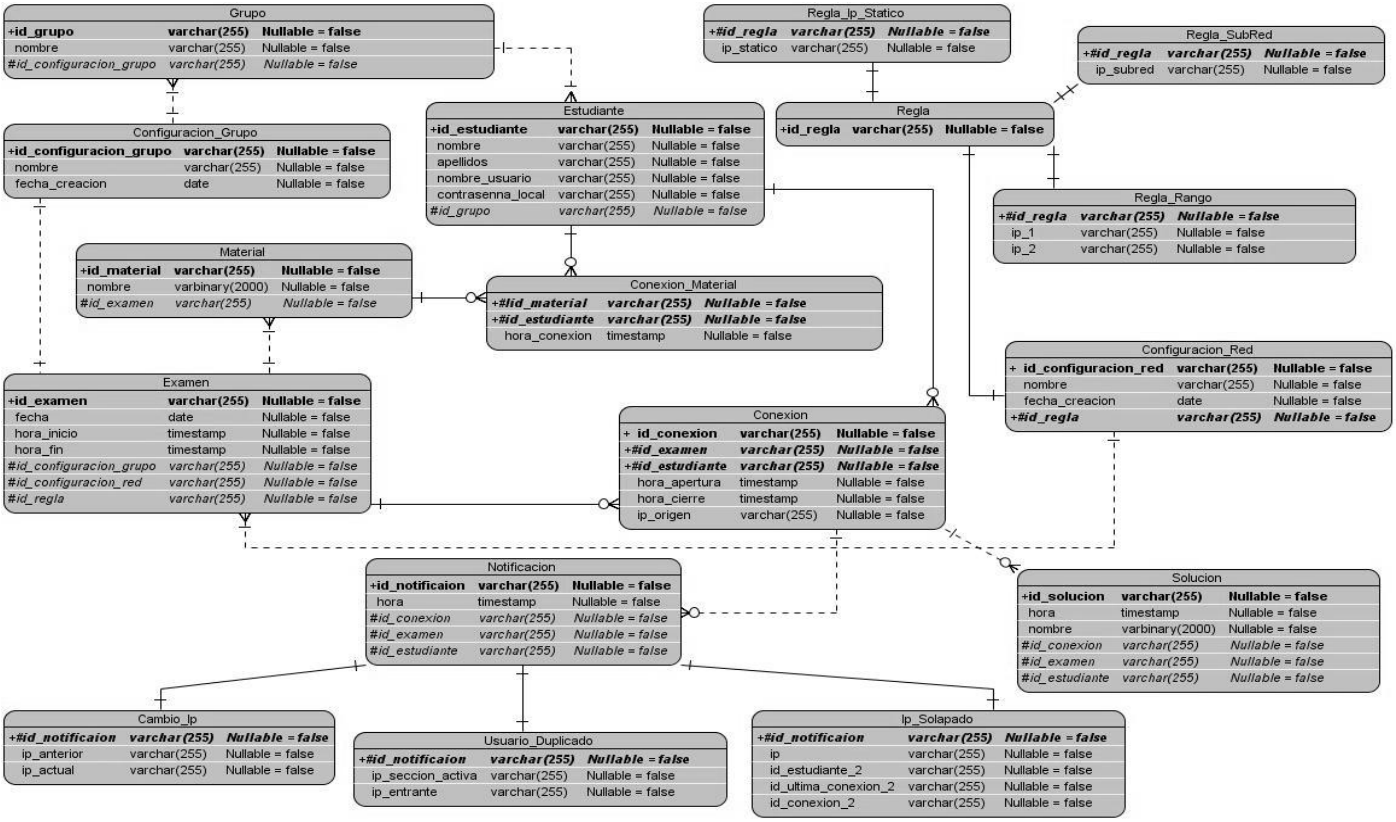


Figura 12 Modelo de datos

3.9 Vista de Despliegue: diagrama de despliegue.

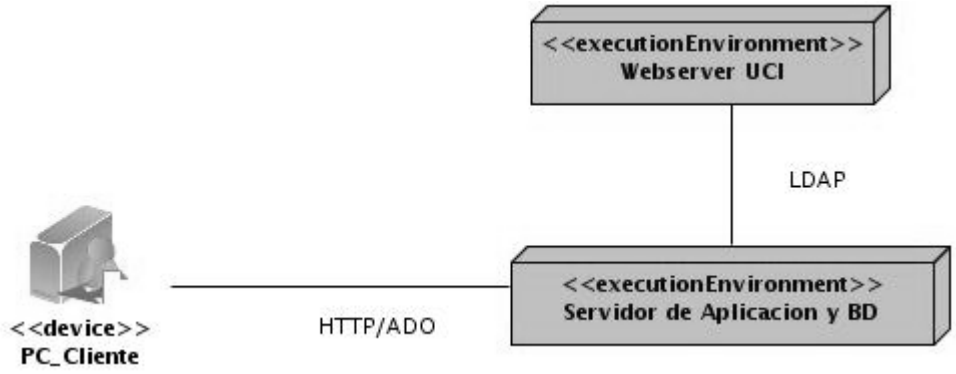


Figura 13 Diagrama de despliegue.

3.10 Conclusiones generales del capítulo

En este capítulo quedo definido el diseño del sistema donde se muestran los artefactos generados en el flujo de trabajo de diseño, además se incluye una breve descripción de los mismos, con el objetivo de que se comprenda perfectamente los requisitos del software. Posibilitando la correcta transformación de los mismos a un diseño que indique como debe ser implementado el software. También se encuentra información referente a los patrones utilizados, a la arquitectura y sus relaciones describiendo las mismas en términos de diagramas de clases.

Conclusiones Generales

Como resultado del trabajo efectuado se realizó el análisis y diseño de la herramienta propuesta para planificar y controlar la ejecución de las evaluaciones prácticas docentes lo cual permitirá su posterior implementación. Como resultados del análisis se obtuvieron los diagramas de colaboración y los de las clases del análisis. Para la realización del diseño se tuvieron en cuenta algunos patrones para modelar el sistema, lo que permitió generar artefactos empleando buenas prácticas, necesarias para desarrollar un software con mayor robustez. Entre los artefactos generados en el diseño se pueden mencionar: diagramas de secuencia, diagramas de clases del diseño ambos, imprescindibles para el buen desarrollo de una solución informática.

De forma general los resultados obtenidos a partir del análisis y diseño del sistema son positivos, tomando como referencia la aplicación de métodos y métricas para validar la solución del mismo. Se realizó el análisis de los requisitos funcionales y no funcionales del sistema informático a desarrollar, se identificaron las funcionalidades que tendrá la aplicación, se diseñó el sistema analizado, obteniéndose los diagrama de Clases del Análisis, así como la estructura y descripción del Modelo de Caso de Uso del Sistema.

Recomendaciones

Profundizar en el estudio de los Espacios Virtuales de Aprendizaje para una futura versión ampliada del sistema.

Realizar los restantes flujos de trabajo que propone la metodología utilizada, llegando a implementar las funcionalidades que hemos propuesto para la herramienta de planificación y control de las evaluaciones prácticas docentes, lo cual tributará a un incremento de la eficiencia y calidad del Sistema Evaluativo en la Universidad de las Ciencias Informáticas.

Glosario de Términos

- CASE: Computer Aided Software Engineering (Ingeniería de Software Asistida por Computación).
- IDE: Integrated Development Environment (Ambiente de desarrollo integrado).
- Interbase: Es un Sistema de Administración de Base de Datos Relacionales (RDBMS).
- Navegador: Programa utilizado para ubicar y ver páginas Web. Por ejemplo Netscape,
- Mosaic, Microsoft Internet Explorer, FireFox, Opera y otros.
- PHP: Es un acrónimo recursivo que significa “Hypertext Pre-processor”. Es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas Web dinámicas.
- Release: Versión de un sistema de software que se libera.
- RUP: Rational Unified Process (Proceso Unificado de Desarrollo de Software).
- Script: Segmento de instrucciones que realizan operaciones sencillas dentro del código de una aplicación.
- TI: Tecnologías de la Investigación (TI).
- UCI: Universidad de las Ciencias Informáticas.
- UML: Lenguaje Unificado de Modelado.
- Web browser: Navegador

Referencia Bibliográfica

ECollege. [Online] [Cited: Febrero 11, 2010.] http://es.wikipedia.org/wiki/ECollege.

EVA. [Online] [Cited: Febrero 11, 2010.] http://eva.uci.cu/mod/resource.

HIBERNATE. 2006.. Hibernate Reference Documentation . [Online] 2006. http://www.hibernate.org.

KASEYA, A. S. G. 2009. Kaseya Enterprise Edition. [Online] 2009. [Cited: febrero 12, 2010.] http://www.kaseya.com.mx/products/remote-software-deployment.php.

2006. Moodle. *Moodle.* [Online] febrero 15 , 2006. [Cited: Febrero 11, 2010.] http://docs.moodle.org/es/P%C3%A1gina_Principal.

Apache-Tomcat. 2009 . Apache-Tomcat. [Online] 2009 . http://es.wikipedia.org/wiki/Apache-Tomcat.

Morgan, G. 2003. Faculty Use of Course Management Systems. [Online] 2003. [Cited: Febrero 11, 2010.] http://www.educause.edu/ir/library/pdf/ers0302/rs/ers0302w.pdf.

MySQL. 2009 . MySQL. [Online] 2009 . http://es.wikipedia.org/wiki/MySQL.

PARADIGM, VISUAL. 2009. VISUAL, PARADIGM. [Online] 2009. http://www.visual-paradigm.com/product/vpuml/.

PHP. 2009 . [Online] 2009 . http://es.wikipedia.org/wiki/.php.

SOFTWARE. 2007 . Rational Software. [Online] 2007 . www.rational.com.

PostgreSQL. 2007. PostgreSQL. [Online] 2007. http://www.http- peru.com/postgresql.php

Bibliografía

ANDRES. 2004. *Extreme Programming Explained: Embrace Change (Paperback).2da edición.* 2004.

BRAVO SANTOS CRESCENCIO Y GARCIA RUBIO, FELIX OSCAR. *Ingeniería del Software. Metodologías de Desarrollo de Software.*

ESCRIBANO. 2002. *Introducción a Extreme Programming.* 2002.

FERNANDEZ PEREZ YUDIT. *Enfoque administrativo de la Ingeniería de Software, 2007.*

JACOBSON, IVAR and RUMBAUGH, JAMES. 2000. . *El Proceso Unificado de Desarrollo de Software. Madrid : s.n., 2000.*

JOHNSON. 2004. *Expert One on one J2EE Development Without EJB.* 2004.

LARMAN. 1999. *UML Y Patrones. Introducción al análisis y diseño orientado a objeto.* México : s.n., 1999.

M.PANIAGUA. 2005. 2005.

MENDOZA SANCHEZ MARIA A. *Metodologías De Desarrollo De Software. Informatízate. 2004.*

ÖVERGAARD, GUNNAR. 2004. *Use Cases Patterns and Blueprints.* 2004.

PRESSMAN, R. S. 2005.. *Ingeniería de Software. Un enfoque Práctico.* 2005.