Universidad de las Ciencias Informáticas

Dirección de Informatización

# ANÁLISIS Y DISEÑO DEL MÓDULO PROFESOR DEL SISTEMA DE GESTIÓN ACADÉMICA AKADEMOS.

Trabajo de diploma para optar por el título de:

**Ingeniero en Ciencias Informáticas**

Akademos
Gestión académica

Autores: Inalvys Hernández Manso
Alexey Peña Paz

Tutora: Ing. Olivia Rodríguez Abril

Ciudad de la Habana, Junio 2008

*No hacen falta alas*
*para hacer un sueño*
*basta con las manos*
*basta con el pecho*
*basta con las piernas*
*y con el empeño.*

*Silvio Rodríguez*

# DEDICATORIA

*A nuestros padres,*

*por ayudarnos a hacer realidad nuestros sueños.*

# AGRADECIMIENTOS

A nuestros padres y hermanos por todo el amor, la confianza y el apoyo que nos han brindado siempre.

A la Revolución cubana y en especial a Fidel por darnos la oportunidad de formarnos en la Universidad del futuro y ser parte de sus sueños.

A la UCI, por brindarnos la oportunidad de ser mejores.

A los profesores que contribuyeron a nuestra formación como profesionales.

A nuestras familias y vecinos por hacernos sentir queridos.

A Elsa y a Vicky, por todo su cariño.

A todos los amigos que hemos ganado a lo largo de esta carrera, gracias por compartir nuestras tristezas y alegrías.

A “negri” por el amor incondicional en los días más difíciles.

A los que han puesto un granito de arena para el desarrollo de este trabajo, en especial a nuestra tutora Olivia por todo su apoyo.

A Damian, Ruben y Yanet, por estar siempre dispuestos a ayudar.

# 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 _____.

Inalvys Hernández Manso

_____

Alexey Peña Paz

_____

Ing. Olivia Rodríguez Abril

_____

# OPINIÓN DEL TUTOR

Título: **Análisis y diseño del módulo Profesor del Sistema de Gestión Académica Akademos.**

Autores: **Inalvys Hernández Manso y Alexey Peña Paz.**

La tutora del presente Trabajo de Diploma considera que durante su ejecución los estudiantes mostraron las cualidades que a continuación se detallan.

El trabajo realizado se desarrolló con muy alta independencia y responsabilidad por parte de los diplomantes, los cuales demostraron en todo momento estar capacitados para asumir y acometer correctamente el trabajo propuesto. Su labor fue desempeñada de forma ardua y constante lo que le permitió desenvolverse fácilmente y avanzar sin grandes contratiempos.

El trabajo cuenta además con una alta originalidad, creatividad y calidad científico-técnico. Se tuvieron en cuenta las tendencias actuales de migración hacia el software libre, lo cual se evidencia en el lenguaje y herramientas utilizados para el desarrollo del análisis y diseño.

Como elemento a resaltar, se señala la utilización de patrones y de un framework de desarrollo, quedando plasmado en el documento de tesis, cuyo contenido debe constituir una referencia válida para aquellos que pongan en práctica y explotación el estudio obtenido como resultado.

Nuestra institución es la mayor beneficiada con el resultado de este trabajo, pues mejora de forma sustancial la gestión de los profesores en la Universidad.

Por todo lo anteriormente expresado considero que los estudiantes están aptos para ejercer como Ingenieros en Ciencias Informáticas; y propongo que se le otorgue al Trabajo de Diploma la calificación de _ puntos, considerando que los resultados deben ser publicados y presentados en eventos científicos.

**Ing. Olivia Rodríguez Abril**

13 de junio de 2008

Firma

# RESUMEN

La Universidad de las Ciencias Informáticas está llamada a ser, por sus características excepcionales, una universidad de excelencia y élite en el desarrollo de software y la informatización de la sociedad cubana. En la actualidad la UCI cuenta con un grupo de proyectos productivos e investigativos encaminados a dar solución a las principales problemáticas que presenta el país y la propia institución en materia de informatización. Akademos es uno de los proyectos que se llevan a cabo y que se encarga de optimizar los procesos de gestión académica de la Universidad a través de un sistema automatizado. Este sistema se encuentra actualmente en plena utilización, pero a pesar de las facilidades que ofrece se han detectado algunas dificultades que atentan contra su buen funcionamiento e impiden que el mismo pueda ser utilizado por otras entidades educativas, por lo que en estos momentos se encuentra sometido al desarrollo de una nueva versión. En el presente trabajo se estudian los procesos que se llevan a cabo en los centros educacionales para la gestión y la planificación de los docentes, que es una de las áreas con que cuenta este sistema actualmente. Se propone como solución el diseño de un módulo que automatice todos los procesos que intervienen en la gestión de los profesores. Para el desarrollo del trabajo se utiliza como metodología de desarrollo RUP, UML como lenguaje de modelado, Visual Paradigm como herramienta CASE, Symfony como framework de desarrollo y PHP como lenguaje de programación.

# ÍNDICE

**INTRODUCCIÓN ..... 1**
**CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA ..... 8**
1.1 Introducción ..... 8
1.2 Sistemas informáticos para la gestión académica ..... 8
1.2.1 Sistemas de gestión académica en el mundo ..... 8
1.2.2 Sistemas de gestión académica en Cuba ..... 13
1.2.3 Sistemas de gestión académica en la UCI ..... 16
1.2.4 Conclusiones del estudio realizado ..... 18
1.4 Metodologías para el desarrollo de software ..... 19
1.5 Herramientas para el desarrollo de software ..... 24
1.6 Framework de desarrollo ..... 26
1.7 Lenguaje de programación ..... 27
1.8 Sistema gestor de base de datos ..... 28
1. 9 Conclusiones ..... 29
**CAPÍTULO 2: DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA ..... 30**
2.1 Introducción ..... 30
2.2 Principales reglas del negocio a considerar ..... 30
2.3 Actores del negocio ..... 30
2.4 Trabajadores del negocio ..... 31
2.5 Diagrama de casos de uso del negocio ..... 31
2.6 Descripciones textuales de los casos de uso del negocio ..... 32
2.7 Técnicas de recopilación de requisitos ..... 36
2.7.1 Requerimientos funcionales ..... 38
2.7.2 Requerimientos no funcionales ..... 49
2.8 Actores del sistema ..... 55
2.9 Diagrama de casos de uso del sistema ..... 55
2.10 Descripciones abreviadas de los casos de uso del sistema. ..... 56
2.11 Conclusiones ..... 59
**CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA ..... 60**
3.1. Introducción ..... 60

3.2 Análisis ..... 60
3.2.1 Diagrama de clases del análisis. ..... 61
3.2.2 Diagramas de colaboración ..... 63
3.3 Diseño ..... 64
3.3.1 Patrones de diseño ..... 64
3.3.2 Diagrama de clases del diseño ..... 69
3.3.3 Diagramas de secuencia ..... 75
3.4 Diseño de la base de datos ..... 77
3.4.1 Modelo lógico ..... 77
3.4.2 Modelo físico ..... 78
3.5 Diagrama de despliegue ..... 79
3.6 Conclusiones ..... 79
**CAPÍTULO 4: ANÁLISIS DE COSTO ..... 80**
4.1 Introducción ..... 80
4.2 Estimación del esfuerzo ..... 80
4.3 Conclusiones ..... 86
**CONCLUSIONES ..... 87**
**RECOMENDACIONES ..... 88**
**BIBLIOGRAFÍA REFERENCIADA ..... 89**
**BIBLIOGRAFÍA CONSULTADA ..... 91**
**GLOSARIO DE TÉRMINOS ..... 95**

# ÍNDICE DE TABLAS

Tabla 2.1 Actores del Negocio. ..... 31
Tabla 2.2 Trabajadores del Negocio ..... 31
Tabla 2.3 Descripción de CU Ubicar Profesor ..... 32
Tabla 2.4 Descripción CU Dar Baja Profesor ..... 33
Tabla 2.5 Descripción CU Reasignar Grupos Profesor ..... 35
Tabla 2.6 Actores del sistema ..... 55
Tabla 2.7 Resumen del CU Gestionar Profesor ..... 56
Tabla 2.8 Resumen del CU Buscar Profesor ..... 57
Tabla 2.9 Resumen del CU Gestionar Profesor Grupo ..... 57
Tabla 2.10 Resumen del CU Gestionar Tipo Profesor ..... 57
Tabla 2.11 Resumen del CU Gestionar Categoría Docente ..... 58
Tabla 2.12 Resumen del CU Gestionar Categoría Científica ..... 58
Tabla 2.13 Resumen del CU Consultar información Profesor ..... 59
Tabla 3.1 Factor de peso de los actores sin ajustar ..... 81
Tabla 3.2 Factor de peso de los casos de uso sin ajustar ..... 81
Tabla 3.3 Factor de complejidad técnica. ..... 82
Tabla 3.4 Factor ambiente. ..... 83
Tabla 3.5 Distribución del esfuerzo ..... 85

# ÍNDICE DE FIGURAS

Figura 1 Diagrama de CU del negocio ..... 32
Figura 2 Diagrama de casos de uso del sistema ..... 56
Figura 3 Diagrama de clases del análisis CU Gestionar Profesor ..... 61
Figura 4 Diagrama de clases del análisis CU Buscar Profesor ..... 61
Figura 5 Diagrama de clases del análisis CU Gestionar Categoría Docente ..... 62
Figura 6 Diagrama de clases del análisis CU Gestionar Tipo Profesor. ..... 62
Figura 7 Diagrama de clases del análisis CU Gestionar Categoría Científica. ..... 63
Figura 8 Diagrama de colaboración Buscar Profesor ..... 64
Figura 9 Diagrama de clases del diseño CU Gestionar Profesor ..... 70
Figura 10 Diagrama de clases del diseño CU Buscar Profesor ..... 71
Figura 11 Diagrama de clases del diseño CU Gestionar Categoría Docente ..... 72
Figura 12 Diagrama de clases del diseño CU Gestionar Tipo Profesor ..... 73
Figura 13 Diagrama de clases del diseño CU Gestionar Categoría Científica ..... 74
Figura 14 Diagrama de secuencia Buscar Profesor ..... 76
Figura 15 Modelo lógico de datos. ..... 77
Figura 16 Modelo físico de datos ..... 78
Figura 17 Diagrama de despliegue. ..... 79

# INTRODUCCIÓN

El creciente auge del uso de las tecnologías y las soluciones informáticas en la actualidad han generado un grupo de sistemas dirigidos a automatizar procesos vitales dentro de la sociedad con el objetivo de satisfacer las necesidades de todas sus esferas. La utilización ordenada y masiva de las tecnologías de la información y las comunicaciones (TIC) en la vida cotidiana es una tendencia mundial que persigue lograr más eficiencia y simplicidad en todos los procesos que a su vez se revierten en mayor competitividad en los servicios y mayor calidad en la vida de los ciudadanos.

Cuba no se encuentra aislada de estas políticas incluso teniendo en cuenta el reto que constituye en un país subdesarrollado y fuertemente bloqueado este propósito. Desde los primeros momentos de la Revolución se percibió la importancia del papel de la ciencia y la técnica para el desarrollo económico de la nación y con este objetivo se han adoptado medidas para lograr, de forma paulatina, una cultura general en estas prácticas con la intención de llegar a dominarlas e introducirlas en cada sector de la sociedad; siempre teniendo en cuenta los principios y valores de nuestro sistema social y con una marcada tendencia a defender las conquistas logradas.

Una de las fortalezas de nuestro sistema social es la educación. Este sector de la sociedad, que el 1 de enero de 1959 fue declarado de carácter gratuito y democrático, es el encargado de formar a todos los niños y jóvenes en edad escolar y brindar múltiples facilidades a los jóvenes y adultos que están en disposición de continuar estudios, y tiene la inmensa responsabilidad de preparar a las nuevas generaciones y elevar el nivel cultural y profesional de los ciudadanos. Es por ello que la existencia de sistemas informáticos encaminados a facilitar el trabajo en este sector tiene una justificada importancia en nuestro país.

En los centros educacionales la gestión académica se identifica como un proceso medular y que define el funcionamiento del centro y por ende sus resultados. En este ámbito las soluciones informáticas han tenido su incursión y se han desarrollado diversos sistemas encaminados a dar solución de forma automatizada a aspectos

elementales como la matrícula de los estudiantes y la gestión del proceso docente. El objetivo principal de estos sistemas ha sido proveer a los centros estudiantiles de mecanismos que permitan de una forma eficiente la planificación, organización y control de sus actividades académicas de manera simplificada.

El Sistema de Gestión Académica Akademos, fue concebido para automatizar el proceso docente de la Universidad de las Ciencias Informáticas. Para lograr este propósito, el sistema cuenta con un grupo de módulos los cuales ofrecen funcionalidades específicas logrando automatizar los principales elementos que intervienen en la gestión docente del centro de estudios. En la actualidad, Akademos ha sido desplegado en la sede central, facultades regionales de la UCI y en el Instituto Superior del MININT Eliseo Reyes Rodríguez "Capitán San Luis".

Aunque Akademos ha tenido una excelente acogida en la universidad por los disímiles procesos que una vez automatizados han permitido un mejor funcionamiento del centro en cuanto a la actividad docente, aún existen inconformidades en algunos aspectos que al ser reformados aumentarían considerablemente su valor a la vez que perfeccionaría el trabajo para lograr mayor eficiencia y acercarlo más a las necesidades actuales de la entidad. Se puede señalar además, que la utilización de Akademos en el Instituto Superior del MININT, ha permitido identificar un conjunto de problemáticas que hacen del mismo un sistema imposible de aplicar en otras entidades educacionales además de la UCI.

Por todo lo anteriormente descrito se propone desarrollar la segunda versión del Sistema de Gestión Académica Akademos, con el objetivo de perfeccionar las funcionalidades que automatiza a través de cada uno de sus módulos teniendo en cuenta los requerimientos actuales de los clientes. Además, incluir un grupo de acciones que debe soportar para responder a las necesidades existentes de forma tal que se obtenga un software flexible, configurable y de fácil utilización, así como la migración total del sistema hacia plataformas de Software Libre.

El claustro docente ocupa un lugar significativo dentro de cualquier centro de estudios ya que es el máximo encargado de liderar, dirigir y formar a la gran masa estudiantil. Por ello es de gran importancia que en los sistemas académicos se conciba un espacio dirigido a la gestión de los mismos y que permita al centro mantener un control actualizado de todos sus datos y las actividades que desempeñan.

Entre los módulos con que cuenta el actual sistema Akademos, se encuentra el Módulo Profesor. El mismo brinda la posibilidad de ubicar a un profesor en una facultad del centro de estudios una vez que ha sido contratado por la Universidad. La ubicación del profesor consiste en la asignación de la asignatura que impartirá y los grupos correspondientes, indicando en cada caso, el plan de estudios por el que se debe impartir la asignatura y el momento del curso escolar en que será acreditada. Indiscutiblemente esta es una de las necesidades que presenta la universidad en cuanto a la gestión de los profesores vinculados a la actividad docente, de manera que una vez automatizada se logró minimizar el tiempo dedicado a ese trabajo en cada una de las facultades y los riesgos de cometer errores tan comunes a la hora de realizar labores engorrosas de forma manual. Aunque no es menos cierto también, que el hecho de que el módulo sólo conciba esta característica del proceso de la gestión de los profesores, es una solución muy primaria teniendo en cuenta los intereses más abarcadores del centro en esta área.

El actual módulo presenta diversos inconvenientes que atentan contra su calidad y su reutilización en sistemas posteriores o nuevas versiones del mismo sistema, tanto desde la óptica del usuario final como de los desarrolladores del software.

Entre los principales problemas que se presentan actualmente, desde la perspectiva del beneficiario, podemos señalar que el módulo no permite la obtención de un grupo de reportes relativos a la planificación docente de la actividad de los profesores que son de importancia significativa para los directivos de cualquier centro de estudio y que hasta el momento se han tenido que realizar de manera manual; también el módulo, por sí mismo, carece de un buscador que permita simplificar el trabajo a la

hora de realizar las asignaciones a los profesores que se ocurren en los diferentes momentos del curso escolar.

Desde el punto de vista de los desarrolladores del software, el módulo presenta una dificultad de orden crítico y es que no posee ninguna documentación consistente que respalde su desarrollo y que permita la retroalimentación a nuevos miembros del equipo. Esto implica diversas complicaciones a la hora de realizar otras iteraciones o desarrollar nuevas versiones del sistema, igualmente hace difícil su mantenimiento puesto que requiere de mucho más tiempo y mayor cantidad de personas para analizar el funcionamiento del módulo una vez que ha sido desplegado.

Además, el hecho de que no exista documentación evoluciona en riesgos más complejos respecto a la calidad del desarrollo; como puede ser el obtener una aplicación que no satisfaga los requerimientos reales de los usuarios debido a la imposibilidad de controlar y revisar periódicamente el trabajo que se desarrolla con el objetivo de corregirlo y perfeccionarlo.

De manera general se puede decir que el módulo fue desarrollado para dar solución a una de las peculiaridades del proceso de gestión de los profesores en la UCI, pero no satisface todas las necesidades que presentan las instituciones educacionales en materia de gestión. El mismo está desarrollado sobre tecnologías propietarias, lo que impide su integración con la nueva versión del sistema contemplada hacia la migración a software libre, todo esto hace imposible la reutilización del módulo actual.

Teniendo en cuenta la situación planteada, el **problema** a resolver queda expresado en la siguiente interrogante: ¿Cómo desarrollar el Módulo Profesor para la segunda versión del sistema Akademos, de forma tal que satisfaga las necesidades en cuanto a la gestión de los profesores?

Para dar solución a la problemática planteada el **objeto de estudio** lo constituye el Módulo Profesor del Sistema de Gestión Académica Akademos y el **campo de acción** es la gestión de los profesores vinculados a la actividad docente.

El **objetivo general** de esta investigación es: elaborar el análisis y diseño del Módulo Profesor para la nueva versión del Sistema de Gestión Académica Akademos.

La **idea a defender** que se plantea es que el correcto desarrollo del análisis y diseño de los procesos de gestión de los profesores permitirá proponer una solución para la nueva versión del Módulo Profesor, que asegure el buen funcionamiento del mismo y la correcta integración con los demás módulos del sistema para su nueva versión.

Para dar cumplimiento al objetivo planteado se propone la realización de las siguientes **tareas:**

- Fundamentar las metodologías y herramientas a utilizar para el desarrollo del trabajo.
- Caracterizar algunos de los sistemas existentes en Cuba y el Mundo para la gestión académica haciendo énfasis en la gestión de los profesores.
- Estudiar los procesos de gestión de los profesores en los centros educacionales.
- Determinar las funcionalidades y restricciones que el módulo debe cumplir.
- Realizar el análisis y diseño de un nuevo módulo para la gestión de los profesores vinculados al proceso docente.

**Métodos científicos de investigación**

Método histórico – lógico: Se pretende analizar a partir del Sistema de Gestión Académica Akademos, el módulo actual de profesores existente en toda su trayectoria de desarrollo para llegar a un entendimiento de su funcionamiento.

Método de la modelación: Con la modelación se podrá conocer más a fondo el objeto que se estudia a partir de la correspondencia que debe establecerse entre el modelo y el objeto en cuestión. La misma permitirá predecir las respuestas del proceso de gestión de los profesores ante posibles incidencias, sin tener que ejecutar el proceso en la realidad.

Método sistémico: Se estudiará el Módulo Profesor con el objetivo de conocer sus relaciones con el medio teniendo en cuenta su estructura interna. Para esto se determinan todos sus componentes y se establecen las relaciones entre los mismos, no sólo de forma estructural sino también de forma dinámica, lo que permitirá esclarecer el comportamiento del sistema como un todo.

Método dialéctico: Se pretende encontrar las contradicciones existentes entre el actual Módulo Profesor y las necesidades existentes en cuanto a la gestión de los docentes y a partir de las mismas explicar y justificar los cambios que se deben producir dando paso a un nuevo módulo.

***Métodos empíricos***

Método de la observación: Se precisa realizar una observación directa, en distintos momentos del desarrollo de la investigación, sobre los procesos de gestión de los profesores que permita conocer la realidad sobre su funcionamiento.

Método de la entrevista: Se realizarán diferentes entrevistas con el objetivo de obtener información sobre el proceso de gestión de los profesores en los centros educacionales y conocer las dificultades presentadas con el actual módulo.

El trabajo se encuentra estructurado en cuatro capítulos.

**Capítulo 1. Fundamentación teórica**

Se realiza un estudio valorativo sobre algunos de los sistemas informáticos vinculados a la gestión académica utilizados en Cuba y en otros países del mundo, profundizando en la gestión de los docentes. Se fundamenta la metodología y herramienta CASE a utilizar para dar solución al problema que se enfrenta y se caracteriza el lenguaje de programación, framework y sistema gestor de base de datos propuestos para el desarrollo del trabajo.

**Capítulo 2. Descripción de la solución propuesta**

Se hace una propuesta de solución a partir de la investigación realizada sobre los procesos de negocio para la gestión de los profesores. Se explican las técnicas de recopilación de requisitos utilizadas para la obtención de los mismos, se especifican los requisitos funcionales y no funcionales que el módulo debe cumplir para lograr su buen funcionamiento y se describen en términos de casos de usos.

**Capítulo 3. Construcción de la solución propuesta**

Aborda aspectos relacionados con la construcción de la solución propuesta, se modelan los diagramas de clases del análisis y del diseño dejando todo listo para la etapa de implementación. Se hace referencia a la arquitectura y patrones utilizados para la construcción de la propuesta de diseño. Se muestra el modelo lógico y físico de datos, así como el diagrama de despliegue.

**Capítulo 4. Análisis de costos**

Se realiza una estimación del esfuerzo necesario para la elaboración del módulo a partir del método de estimación por puntos de casos de uso, el cual permite predecir el tamaño de un sistema a partir de las características de sus requisitos expresados en los casos de uso.

# CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

## 1.1 Introducción

En este capítulo se realiza una investigación referente a los Sistemas de Gestión Académica que se han utilizado en las universidades de nuestro país y específicamente, en la Universidad de las Ciencias Informáticas, así como algunos sistemas relevantes a nivel internacional; realizando un análisis más profundo que contemple la gestión de los profesores para cada uno de ellos. Se describen además las características del lenguaje de programación, sistema gestor de base de datos, herramientas y metodología de desarrollo propuestas para dar solución al problema.

## 1.2 Sistemas informáticos para la gestión académica

Debido a la creciente incorporación de las nuevas tecnologías de la información se han desarrollado sistemas que tienen como tarea fundamental automatizar la gestión académica, la cual constituye uno de los elementos fundamentales para asegurar el buen funcionamiento de cualquier centro de estudios. Estos sistemas generalmente abarcan todo lo relacionado con la matrícula de los estudiantes, el control de las evaluaciones y asistencias, así como otros procesos importantes. Constituyen una poderosa herramienta de trabajo que permite hacer más eficientes los procesos, y optimizar los recursos del centro.

### 1.2.1 Sistemas de gestión académica en el mundo

**SIGA: Sistema Integrado de Gestión Académica**

SIGA es la aplicación informática utilizada por la Universidad de Córdoba (España) para la automatización de la gestión académica. Las funcionalidades soportadas por SIGA son muy amplias, abarcan desde la gestión del acceso a la universidad pasando por la matrícula hasta la tramitación de los títulos, expedientes académicos, estadísticas, informes, gestión de becas y convalidaciones. (1) La versatilidad y flexibilidad del sistema permiten organizar la informatización de diversas instituciones que abarca a conservatorios, academias, colegios, institutos y universidades, así como centros de formación de empresas, y eventos especiales como maestrías y

cursos de postgrado adaptándose de una forma rápida y fácil a las necesidades correspondientes de cada centro. El sistema cuenta con los siguientes módulos:

- Módulo General.
- Módulo de Definición de Estudio.
- Módulo de Alumnos.
- Módulo de Opciones de Alumno. (Este a su vez cuenta con 4 sub-módulos: *Gestión de Tutorías, Gestión de Mensajería, Gestión de Asistencias y Gestión de Calificaciones*).
- Módulo Profesores.
- Módulo Inventario.
- Módulo Ingresos-Gastos.
- Módulo GENFOR.
- Módulo Generador de Horarios.
- Módulo Generador de Diplomas.
- Módulo de Utilidades.

Cada uno de estos módulos cumple una función dentro del sistema y a través de su integración proveen al mismo de la robustez suficiente para ser considerado como un software flexible y eficiente.

En SIGA existe un módulo destinado a la gestión de los profesores que permite al personal de secretaría docente el control de los mismos. Este módulo permite dar alta o baja a un determinado profesor del centro de estudios, brinda la posibilidad de modificar sus datos y mantiene el control de los profesores que actualmente prestan servicio en el centro. A través del sistema se les puede asignar a los profesores el curso, asignatura, grupo y turno en el que impartirá clases, además del plan de estudio. Por otra parte, el sistema cuenta con herramientas que gestionan la información acerca de las incidencias presentadas con el profesor como las faltas a clases; mantiene un fichero con el histórico del centro y permite la elaboración de listados según determinados parámetros que se especifiquen.

**AGORA: Gestión Integral de Academias y Centros Docentes**

Es un producto estándar de gran calidad que permite la gestión académica en cualquier centro docente tanto en academias, colegios, institutos, escuelas de negocios, centros de impartición de máster, formación ocupacional y continua, centros universitarios, centros de idiomas, oposiciones, informática, música, etc. (2) El software cuenta con un estricto control de acceso que permite, una vez que se definen para cada usuario los niveles a los que está autorizado, restringir el acceso a determinadas partes del sistema. Se encuentra disponible en tres ediciones distintas que se adaptan a sus distintos requerimientos o perfiles: AGORA Básico, AGORA Profesional y AGORA Empresarial. AGORA permite realizar diferentes tipos de gestión en un centro de estudios:

- *Gestión Económica:* Permite gestionar los recibos, facturas, cobros, contabilidad, rentabilidad, gastos, pagos a profesores, etc.
- *Gestión Administrativa:* Permite la gestión de expedientes, matrículas, impresión de documentos, carné de estudiante, biblioteca, correspondencia enlazada, etc.
- *Gestión Comercial:* Permite el seguimiento de contactos y de avisos comerciales así como el tratamiento con empresas, presupuestos, envío de correos automatizado a clientes, etc.
- *Gestión Docente:* Permite gestionar los horarios docentes, profesores, asignaturas, aulas, control de asistencias y docencias, calificaciones, diplomas, etc.

AGORA provee al centro de mecanismos que facilitan su gestión académica de forma automatizada entre los cuales se destacan por su importancia los siguientes: a través del sistema se puede mantener en el centro de estudio el control de los alumnos tanto los actuales como el histórico, ofrece además el control por separado de los alumnos que están recibiendo docencia y los clientes que las abonan. Este sistema permite también la gestión de los horarios docentes y brinda la información acerca de la disponibilidad de profesores y aulas según el horario. Mantiene el control de la asistencia real, permitiendo generar un registro automatizado de la asistencia al que

se le pueden agregar las incidencias producidas y permite tener de manera fácil el control de las ausencias. Por otra parte, AGORA genera estadísticas con presentación gráfica sobre aspectos relevantes como la procedencia del estudiantado o el progreso de la matrícula; permite además, obtener estadísticas de forma personalizada.

AGORA cuenta con un subsistema para la Gestión Centralizada de Profesores que permite:

- Dar alta a un profesor en el centro de estudios.
- Manipular los datos económicos relacionados con el profesor tales como: el importe fijo mensual que recibe el profesor, el cómputo de liquidaciones de profesores, cálculos de la rentabilidad, etc.
- Dar baja a un profesor tanto temporal como definitiva.
- Vincular a un profesor con las asignaturas que está en disposición de impartir.
- Vincular al profesor con un horario de disponibilidad. (No es su horario de clases, sino de aquel en que está disponible por si surge la ocasión de impartir docencia.)
- Realizar una serie de consultas acerca de un profesor entre las que se encuentran:
    1. Mostrar el listado de alumnos que tiene un profesor.
    2. Mostrar las clases que imparte un profesor.
    3. Mostrar los datos generales del profesor.
    4. Mostrar el listado de asignaturas que puede impartir un profesor.
    5. Mostrar los horarios en los cuales está disponible el profesor y los horarios en los que imparte clases.
    6. Mostrar el listado de grupos que atiende un profesor.

**ALBA: Sistema Informático Abierto de Gestión Unificada para Unidades Educacionales**

ALBA, es un proyecto de desarrollo del Software Libre para la realización de un *"Sistema Informático Abierto de Gestión Unificada para Unidades Educacionales"* que

brinda una herramienta para mejorar el trabajo cotidiano en las escuelas. El objetivo principal del proyecto es crear un sistema de gestión abierto para centros educacionales capaz de adaptarse a las necesidades de la unidad donde se implante. Este sistema ha sido desarrollado a partir de la información recopilada a través de un grupo de entrevistas realizadas a responsables del sector educacional y pretende poder ser utilizado en diferentes niveles de entidades educativas. La primera versión de ALBA fue pensada para realizar pruebas en escuelas primarias de la Ciudad de Buenos Aires y tenía en cuenta principalmente la gestión de las unidades educacionales (establecimientos, ciclos, calendarios, etc.), gestión de alumnos (expedientes, seguimientos, consultas, etc.) y la gestión de docentes (expedientes, horarios, etc.).

En octubre del 2007 fue liberada la versión 1.0 final, incorporando mejoras en los módulos liberados en la versión anterior con el objetivo de beneficiar la funcionalidad de los mismos y tomando como punto de partida las sugerencias de los establecimientos donde se estaba utilizando ALBA y de colaboradores que han sumado ideas y aportes al proyecto. La nueva versión conjuga entre todos sus módulos: la administración de usuarios, gestión de varios establecimientos por organización, gestión de grupos, gestión integral de información del alumno, gestión de docentes, gestión de calendarios y horarios, gestión del espacio, y generación de informes y consultas.

Para el caso de los docentes, debe configurarse previamente en el sistema:

- Tipos de docentes: se establecen las diferentes categorías de docentes según el cargo, proporcionando una breve descripción. Posibilita acciones de edición.
- Motivos de baja: se establecen las posibles motivaciones de las bajas de docentes, por ejemplo: renuncia, retiro, ascenso, etc. En este listado se ingresa el nombre del tipo de baja del cargo y una breve descripción explicativa. Se permiten acciones de edición.

La gestión de los docentes permite mantener la información de los profesores del centro en relación a sus datos personales. Permite además crear la relación existente entre un docente y la actividad/materia que dicta el mismo.

Por su tipo de licenciamiento bajo la GNU/GPL, el sistema permitirá una reutilización y actualización constante, así como la posibilidad de aprovechamiento de módulos ya realizados por otros emprendedores (siempre que sean compatibles con esta licencia). (3) Esta herramienta de gestión fue desarrollada utilizando Symfony como framework de desarrollo para el lenguaje de programación PHP5, con tecnología de programación totalmente Orientada a Objetos.

### 1.2.2 Sistemas de gestión académica en Cuba

**SDI: Sistema Docente Integral**

Este sistema se creó en el Instituto Superior Politécnico “José Antonio Echeverría” (CUJAE) y surgió debido a la vital importancia que tiene para este centro el control, planificación y resultado de los procesos docentes. El SDI permite registrar, procesar y crear mecanismos de recuperación de la información relacionada con el proceso docente, tanto de pregrado como de postgrado de forma automatizada, lo que es una ventaja ya que con su uso se pueden realizar estos procesos de forma eficiente permitiendo optimizar los recursos del centro.

El SDI está conformado actualmente por 8 módulos y se trabaja para aumentar sus funcionalidades. Los módulos que están diseñados son:

- Secretaría de Facultades.
- Matrícula.
- Estadística.
- Archivo Histórico.
- Recuperación de Información.
- Pago de Estipendios.
- Planificación Docente.
- Postgrado. (4)

**GESTACAD: Sistema de Gestión Académica**

GESTACAD es un sistema informativo automatizado, desarrollado por el departamento de informática de la Universidad de Matanzas "Camilo Cienfuegos". Permite actualizar y mantener la información sobre estudiantes y profesores de una Universidad y obtener determinados resultados propios del trabajo de las áreas implicadas, aunque el grueso de las informaciones se obtiene mediante el acceso al sitio WEB de la Universidad.(5)

Este sistema surgió a partir de la necesidad de desarrollar un software que permitiera automatizar la gestión académica de las universidades en Cuba, el cual gestionara una parte de la información académica de los estudiantes universitarios y la información de los profesores que forman parte del proceso docente educativo. En su desarrollo el sistema ha pasado por diferentes etapas desde su concepción hasta el momento actual con el objetivo de mejorar sus funcionalidades lo que ha dado lugar a dos versiones del sistema. La segunda versión ha sido concebida para la migración total del sistema hacia plataformas de software libre, además incluye mejoras y nuevos módulos que no estaban concebidos en la primera versión como Administración, Secretaría y Matrícula. En la actualidad este sistema permite entre otras funcionalidades la búsqueda de un alumno en la base de datos, mostrar el listado de estudiantes por grupo, generar reportes dinámicos de la información existente, reportes de notas por asignatura y grupo, tablas con los resultados docentes de un grupo en un semestre, reportes de los resultados académicos de un estudiante en toda su carrera, actas de exámenes de las diferentes asignaturas, registros de características de un grupo de estudiantes, entre otras. GESTACAD cuenta con los siguientes módulos:

- Módulo Administración.
- Módulo Web para las Secretarías Docentes.
- Módulo Web para los Jefes de Departamentos Docentes.
- Módulo Web para la Gestión de la Matrícula.
- Módulo Web para los Profesores.

En GESTACAD el Módulo Web para los Profesores permite a los mismos llevar el control docente de sus estudiantes, el control de las evaluaciones así como reportes relativos a su carga docente. El módulo Web para Jefes de Departamentos Docentes permite a los mismos, entre otras acciones, la asignación de la carga docente y el control sobre los profesores del Dpto.

**SIGENU: Sistema de Gestión de la Nueva Universidad**

El Sistema de Gestión de la Nueva Universidad (SIGENU) es un software que se ha desarrollado con el fin de ser una herramienta que permita la gestión de toda la información académica vinculada con la educación superior en Cuba. En correspondencia con su carácter nacional y la gran diversidad de sistemas de enseñanza superior con que cuenta la universidad cubana, este sistema ha sido concebido de manera tal que sea capaz de brindar gran seguridad e integridad de la información, y a la vez, ser tan flexible que permita ser adaptado a todos los centros de educación superior del país con sus diversas particularidades y distintas maneras de realizar determinados procedimientos. (6)

SIGENU está compuesto por cuatro elementos fundamentales:

*Base de Datos:* Donde se almacena la información del sistema.

*Servidor de Aplicaciones:* Actúa como intermediario entre la Base de Datos y las aplicaciones clientes y permite dar servicio a través de la red.

*Aplicación Cliente de Administración:* Permite la inserción y actualización de los usuarios y contiene las funcionalidades que permiten a los administradores monitorear el sistema.

*Aplicación Cliente (SIGENU):* Aplicación Desktop que permite la inserción y actualización de toda la información que se registra en el sistema y la obtención de reportes. Consta de los siguientes módulos:

- Codificadores: Contiene toda la información con que debe contar el sistema y que es provista por el Ministerio de Educación Superior (MES).

- Matrícula: Permite realizar el proceso de matrícula a través del cual los estudiantes pasarán a ser registrados en el sistema como estudiantes de Educación Superior.
- Control de estudiantes: Permite buscar un estudiante registrado en el sistema, modificar los datos de un estudiante tanto personales como docentes, ubicar a un estudiante en un grupo o cambiarlo de grupo, realizar el pase de estudiantes a otros años de estudio y definir los que serán repitentes así como dar baja a un estudiante del centro ya sea por licencia de matrícula, resolución o traslado.
- Plan de Estudio: Permite la creación de los planes de estudio para las diferentes carreras del centro así como realizar ajustes y modificaciones a un plan de estudio específico.
- Evaluaciones: Permite registrar, modificar o eliminar las evaluaciones de los estudiantes registrados en el sistema así como premios y bonificaciones.
- Reportes: Permite obtener diversos reportes con los que se puede recuperar toda la información necesaria del sistema.

### 1.2.3 Sistemas de gestión académica en la UCI

Desde los primeros años de creada la Universidad se demostró la necesidad de que existiera un sistema de gestión académica que informatizara los procesos docentes vitales para el buen funcionamiento y eficiencia del centro.

En la universidad han sido utilizados varios sistemas para la gestión académica entre los que se encuentra GESTACAD. Este sistema, al comienzo, fue de mucha utilidad debido a algunas funcionalidades que ofrecía, pero no cumplía todas las expectativas del sistema docente empleado en la UCI, pues no brindaba la posibilidad de contar con más de un plan de estudio ni daba seguimiento a las evaluaciones entre otras inconvenientes, por lo que dejó de tener validez.

Otro de los sistemas usados fue UCIMAT, este sistema permitía controlar algunos de los principales procesos docentes como la matrícula, permitía además, realizar

algunas consultas y reportes, pero sólo los que traía por defecto, de lo contrario, había que ir directamente a la base de datos. Este sistema tampoco era la solución a las necesidades de la Universidad para la gestión de los procesos docentes. La necesidad de un sistema capaz de resolver esta problemática era cada vez mayor teniendo en cuenta el aumento progresivo de la matrícula del centro en cada curso docente. Producto de esta necesidad se crea en el año 2003, el Sistema de Gestión Académico de la Universidad Akademos, con el objetivo de automatizar los procesos que intervienen en la labor académica del centro de estudios y que pueda, de manera constante, enfrentarse a los cambios y adaptarse de forma natural a los nuevos contextos o formas de hacer. Akademos se divide en 7 módulos:

- Módulo Plan de Estudio.
  Este módulo permite la creación del plan de estudio de la carrera definiendo los niveles divididos en momentos que los estudiantes deben vencer una vez concluidas las asignaturas. También permite la definición de los diferentes perfiles y disciplinas en las que se agrupan las asignaturas.
- Módulo Matrícula.
  Este módulo es el encargado de mantener el control de los estudiantes una vez que ingresan al centro así como la gestión de los movimientos que se le aplican a un estudiante durante su paso por la universidad. El módulo además es el encargado de definir la estructura del centro de estudios en la cual se agrupa un conjunto de estructuras organizadas jerárquicamente.
- Módulo Estudiante.
  Este módulo permite al estudiante interactuar con el sistema con el objetivo de consultar el registro de sus evaluaciones.
- Módulo Expediente.
  Este módulo almacena de forma digital un conjunto de información referente al desempeño académico y a otros aspectos del comportamiento de un estudiante en su paso por la universidad.

- Módulo Reportes.
  Este módulo cuenta con diferentes herramientas que permiten al usuario el diseño y publicación de nuevos reportes que involucren diferentes aspectos de los estudiantes.
- Módulo Registro.
  Este módulo es el encargado de permitir a un profesor llevar el control de las asistencias y las evaluaciones aplicadas a sus estudiantes.
- Módulo Profesor.
  Este módulo permite mantener el control de la plantilla de profesores y la carga docente relativa a los mismos que no es más que las asignaturas que imparte en cada uno de los grupos que atiende.

### 1.2.4 Conclusiones del estudio realizado

Se estudiaron sistemas de gestión académica que son utilizados en algunos países del mundo así como otros que se han utilizado en nuestro país obteniendo como resultado que ninguno de los sistemas anteriormente citados puede ser utilizado para la gestión de los profesores en las entidades educacionales cubanas.

En el caso de los sistemas AGORA y SIGA, se puede señalar que a pesar de que ambos cuentan con una estructura que se encarga de la gestión de los profesores, los principales procesos de la gestión académica que automatizan no responden a las necesidades actuales. Otro de los inconvenientes que presentan estos sistemas es que están desarrollados sobre tecnologías propietarias lo que va en contra de las políticas de migración hacia software libre del país y la Universidad.

En el caso del ALBA a pesar de ser un sistema desarrollado sobre software libre y contar con una estructura para la gestión de los profesores del centro, se puede señalar que no es una solución madura aún por lo que no es factible su utilización.

Entre los sistemas que se han utilizado por diferentes universidades del país y que fueron estudiados se puede señalar en el caso del SDI y el SIGENU, que no cuentan

con una estructura que permita gestionar los profesores de un centro de estudio y en el caso de GESTACAD, a pesar de contar con una estructura que automatiza algunos aspectos de la gestión de los profesores, aún le faltan muchos elementos para poder ser aplicado. El sistema Akademos cuenta con un módulo para la gestión de los docentes del centro, pero carece de un grupo de funcionalidades que permitan poder utilizarlo en otros centros de estudio de una forma satisfactoria.

### 1.4 Metodologías para el desarrollo de software

Las metodologías de desarrollo definen un conjunto de procedimientos, técnicas, herramientas y soporte documental que deben ser considerados para la realización de un software. Las metodologías han ido evolucionando históricamente según las necesidades de los clientes, la importancia que ha ido alcanzando el software en la sociedad, así como las diferentes etapas por las que ha transitado el mismo, desde el desarrollo convencional, pasando por el estructurado hasta el desarrollo orientado a objetos. Producto a esto, las metodologías pueden ser agrupadas en dos grandes grupos, Metodologías Estructuradas y Metodologías Orientadas a Objetos. La historia y evolución de ambas va ligada al surgimiento de la programación estructurada y los lenguajes de programación orientada a objetos respectivamente.

En los últimos tiempos la cantidad y variedad de los procesos de desarrollo ha aumentado de forma impresionante, sobre todo teniendo en cuenta el tiempo que estuvo en vigor como ley única el famoso desarrollo en cascada. Se podría decir que en estos últimos años se han desarrollado dos corrientes en lo referente a los procesos de desarrollo, los llamados *métodos pesados* y los *métodos ligeros*. (7) La diferencia fundamental entre ambos es que mientras los métodos pesados intentan conseguir el objetivo común por medio de orden y documentación, los métodos ligeros (también llamados métodos ágiles) tratan de mejorar la calidad del software por medio de una comunicación directa e inmediata entre las personas que intervienen en el proceso. A continuación se estudian algunas de las metodologías más utilizadas a nivel mundial dentro de estas dos corrientes.

**XP: eXtreme Programming**

XP es un método ligero recomendado para utilizar en proyectos cortos y equipos pequeños. XP intenta reducir la complejidad del software y minimizar el riesgo de fallo del proceso por medio de un trabajo orientado directamente al objetivo, basado en las relaciones interpersonales y la velocidad de reacción, por lo que necesita de la presencia permanente de un representante del cliente que esté a disposición del equipo de desarrollo y en condiciones de contestar rápido y correctamente a cualquier pregunta del equipo de forma que no se retrase la toma de decisiones. XP define como base del software a desarrollar *UserStories (*historias del usuario. Estas historias las escribe el cliente y describen escenarios sobre el funcionamiento del software, a partir de estas se crea un plan de *releases* (entregas) entre el cliente y el equipo de desarrollo, definiendo para cada entrega los objetivos que se deben cumplir y las iteraciones (de pocas semanas de duración) necesarias para dar cumplimiento a los mismos. El resultado de cada iteración será entregado al cliente para que lo evalúe y apruebe; en base a su opinión se definen las siguientes iteraciones del proyecto o se adaptará el plan de entregas e iteraciones hasta que el cliente dé su aprobación y el software esté a su gusto. La programación del software se produce en parejas que rotan cíclicamente a lo largo del desarrollo, por lo que cada miembro del equipo trabaja al menos una vez con cada uno de los demás integrantes y con cada componente del software a desarrollar, de forma que el conocimiento de la aplicación completa lo posea el equipo entero y no unos pocos miembros. Otra de las características que posee XP es que el código pertenece al equipo completo por lo que un programador puede cambiar cualquier parte del mismo cuando lo necesite. En XP sólo se programa la funcionalidad que es requerida para la entrega inmediata, siguiendo un diseño evolutivo que trata de conseguir las funcionalidades deseadas de la forma más sencilla posible.

Uno de los principales problemas presentados por la metodología, que hace imposible su utilización para el desarrollo de este trabajo, se encuentra en la necesidad de contar en el equipo de desarrollo con el cliente o con un representante permanente del mismo, esto no podrá ser posible ya que el módulo para la gestión de los profesores debe responder principalmente a las necesidades de los directivos y

secretarias docentes de las facultades en los cuales recae una gran cantidad de trabajo lo que hace imposible que puedan estar a disposición del equipo en todo momento. Otro de los inconvenientes que presenta XP para el desarrollo del trabajo es que es una metodología muy orientada a la implementación y genera muy poca documentación fuera del código fuente, esto es una característica no deseable teniendo en cuenta la importancia que tiene documentar el trabajo desarrollado para las posteriores versiones del módulo o su mantenimiento y para la preparación de nuevos miembros en el equipo.

**FDD: Feature Driven Developmen**

Es un proceso ágil para el desarrollo de sistemas, que se podría considerar a medio camino entre RUP y XP. FDD está pensado para proyectos con tiempo de desarrollo relativamente cortos y se basa en un proceso iterativo con iteraciones cortas que producen un software funcional que el cliente y la dirección de la empresa pueden ver y monitorizar. Las iteraciones se definen en base a las funcionalidades, que son pequeñas partes del software con significado para el cliente. Un proyecto que sigue FDD se divide en 5 fases:

- Desarrollo de un modelo general.
- Construcción de la lista de funcionalidades.
- Plan de releases en base a las funcionalidades a implementar.
- Diseñar en base a las funcionalidades.
- Implementar en base a las funcionalidades.

Las primeras tres fases se desarrollan principalmente en las primeras iteraciones limitándose a un proceso de refinamiento en las demás. Las dos últimas fases absorben la mayor parte del tiempo según va avanzando el proyecto. El trabajo se realiza en grupo, aunque siempre habrá un responsable con mayor experiencia, que tendrá la última palabra en caso de no llegar a un acuerdo. Esto permite que todos los miembros del equipo formen parte del proyecto y que los menos expertos aprendan de las discusiones de los más experimentados. En el proceso de implementar la funcionalidad también se contemplan como partes del mismo la preparación y

ejecución de pruebas, así como revisiones del código, para distribuir el conocimiento y aumentar la calidad, e integración de las partes que componen el software.

Entre los principales problemas que presenta FDD para el desarrollo del trabajo podemos señalar que no define cómo realizar la captura de requisitos y sólo define el proceder a partir del momento en que ya se conocen las funcionalidades que el sistema debe cumplir. Esto es una característica desfavorable puesto que el no realizar la captura de requisitos puede conducir al diseño de un módulo que no sea la solución a los problemas que realmente tiene el cliente, sobre todo cuando el equipo no conoce a fondo el negocio y no está bien identificado con los procesos que se llevan a cabo en el mismo. Otro de los inconvenientes es que genera muy poca documentación a pesar de definirse como un punto intermedio entre la libertad de XP y la rigurosidad de RUP. FDD genera más documentación que XP, donde apenas existe, pero menos que RUP, que intenta documentar todo. La principal problemática que presenta esta metodología es la necesidad de contar en el equipo con miembros que tengan una vasta experiencia y que marquen el camino a seguir desde el principio. Esta exigencia es imposible de cumplir ya que el desarrollo del trabajo se va a llevar a cabo con estudiantes de la universidad que están cursando el quinto año de la carrera y profesores recién graduados que aún no cuentan con la experiencia que requiere la metodología.

**RUP: Rational Unified Process**

RUP es un proceso de desarrollo de software pesado y constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas. RUP está pensado para proyectos y equipos grandes, en cuanto a tamaño y duración y se caracteriza por ser iterativo e incremental, lo que permite mantener un constante refinamiento del software, estar centrado en la arquitectura y ser guiado por los casos de uso. RUP divide el proceso de desarrollo en ciclos de vida obteniendo un producto superior al final de cada uno. Los ciclos se componen de fases y en las mismas llevan a cabo un conjunto de flujos para el desarrollo de todo el proyecto. Las fases terminan con el cumplimiento de un hito, que no es más que un punto de control que permite evaluar el progreso del trabajo y pueden estar divididas en una o varias iteraciones de

tamaño variable según el proyecto. RUP es uno de los procesos más generales de los existentes actualmente, ya que está pensado para adaptarse a cualquier proyecto. Esta metodología intenta reducir la complejidad del software a través de una planificación intensiva del trabajo. Contempla diferentes elementos de planificación como el Plan de desarrollo, Plan de Iteración, Plan de Calidad, etc. los que permiten mantener un control sobre el desarrollo del software. También contempla la gestión de riesgos que permite reconocer problemas y fallos de forma temprana y prevenirlos. La calidad de los artefactos generados será probada durante la totalidad del ciclo de vida del proyecto a través de distintas medidas de calidad, como convenciones, revisiones, auditorías periódicas, pruebas, etc.

Se considera dentro de las metodologías estudiadas como la más óptima para la realización del trabajo, en primer lugar por ser la metodología con la que el equipo de desarrollo se siente más identificado y posee los conocimientos básicos para trabajar con la misma, esto permitirá que se elimine el tiempo que se tendría que dedicar a realizar estudios o pasar cursos para poder trabajar con otra de las metodologías estudiadas. Otra de las características favorables que ofrece RUP para el desarrollo de este trabajo es que permite mantener un software documentado desde sus inicios lo que es favorable para que al incorporarse nuevos miembros al equipo de desarrollo puedan conocer el trabajo realizado a profundidad de una manera rápida y sencilla. RUP permitirá, además, planificar todas las etapas del desarrollo lo que es de gran importancia para el trabajo con equipos grandes y para la realización de proyectos complejos, ya que de esta forma se mantendrá organizado todo el desarrollo y se tendrán identificados los responsables de cada tarea en cada momento, haciendo mucho más ameno y eficiente el trabajo. Su característica de ser iterativo e incremental permitirá ir perfeccionado el software en cualquier momento del desarrollo, esto es favorable teniendo en cuenta que el sistema se está realizando por estudiantes que no poseen ninguna experiencia en la realización de trabajos complejos y no cuentan con un conocimiento profundo sobre los procesos que se pretenden automatizar lo que requiere de mayor tiempo de estudio del negocio y puede implicar que se cometan errores a la hora de comprender las funcionalidades que el sistema debe cumplir para satisfacer el mismo; esta característica de RUP es

conveniente además teniendo en cuenta la variabilidad de las necesidades del cliente lo que puede incurrir en constantes cambios en los requisitos del sistema. RUP tiene la característica de adaptarse a las necesidades del proyecto o institución que lo esté utilizando.

RUP se basa en UML (lenguaje unificado de modelado) como lenguaje de modelado.

**UML: Unified Modeling Language**

Está consolidado como el lenguaje estándar en el análisis y diseño de sistemas de cómputo. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software previo al proceso de escribir el código. UML permite incluir en el modelado del sistema aspectos conceptuales, como los 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.

### 1.5 Herramientas para el desarrollo de software

Actualmente se consideran a las herramientas de desarrollo del software como herramientas basadas en computadoras que asisten el proceso de ciclo de vida de software, consolidadas en la literatura en la forma de ingeniería de software asistida por computadora (CASE, por sus siglas en inglés). Esto es, software que se utiliza para ayudar a las actividades del proceso de software o software que es utilizado para diseñar y para implementar otro software. Permiten automatizar acciones bien definidas, reduciendo también la carga cognitiva del ingeniero de software, quien requiere libertad para concentrarse en los aspectos creativos del proceso. Este soporte se traduce en mejoras a la calidad y la productividad en el diseño y desarrollo. Las herramientas de desarrollo de software automatizan metodologías de software y desarrollo de sistemas y se vinculan con los diferentes conceptos involucrados en el desarrollo.

**Rational Rose:** Es la herramienta CASE que comercializan los desarrolladores de UML y que soporta de forma completa la especificación del UML. Es una de las más

poderosas herramientas de modelado visual para el análisis y diseño de sistemas orientados a objetos y se utiliza para modelar un sistema antes de construirlo.
Cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases. Propone la utilización de cuatro tipos de modelos para realizar un diseño del sistema, utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite crear y refinar estas vistas creando de esta forma un modelo completo que representa el dominio del problema y el sistema de software. Facilita el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información (vista de casos de uso, vista lógica, vista de componentes y vista de despliegue), pero utilizan un lenguaje común para comprender y comunicar la estructura y la funcionalidad del sistema en construcción.

A pesar de ser una poderosa herramienta para el modelado y de que los miembros del equipo de desarrollo se encuentran plenamente identificados y capacitados para trabajar con la misma, no se va a utilizar para el desarrollo del módulo pues es una herramienta propietaria, por lo que su uso va en contra de las políticas del país y de la universidad para el desarrollo de software.

**Visual Paradigm para UML:** Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Esta herramienta está especializada en la ingeniería del software de bases de datos y permite gestionar proyectos muy complejos con gran sencillez. Permite dibujar todos los tipos de diagramas de clases, generar código desde diagramas, generar documentación automáticamente en varios formatos como Web y PDF, y permite el control de versiones lo que hace posible disponer de múltiples versiones del proyecto para cada necesidad. Incluye herramientas muy interesantes para ingeniería inversa de bases de datos, además es una herramienta colaborativa, es decir, soporta múltiples usuarios trabajando sobre el mismo proyecto y brinda la posibilidad de integrarse con los principales IDEs de desarrollo y en múltiples plataformas. Visual Paradigm ofrece un entorno de creación de diagramas para UML 2.0, permite desarrollar el diseño centrado en casos de uso y

enfocado al negocio lo que genera un software de mayor calidad y a través del uso de un lenguaje estándar común a todo el equipo de desarrollo facilita la comunicación.

Esta es la herramienta de modelado que se utilizará para el desarrollo del módulo, pues a pesar de ser una herramienta propietaria la Universidad posee una licencia para el trabajo con la misma. Esta es una de las herramientas más potentes que existe en la actualidad para asistir el proceso de construcción del software. El hecho de ser una herramienta que soporta el trabajo concurrente de varios usuarios es una característica favorable para el trabajo en equipos grandes y proyectos complejos. Otra de las ventajas que presenta la herramienta para este trabajo es que genera código PHP que es el lenguaje de programación que se propone utilizar en la etapa de implementación.

### 1.6 Framework de desarrollo

**Symfony:** Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows. A continuación se muestran algunas de sus características.(8)

Symfony se diseñó para que se ajuste a los siguientes requisitos:

- Fácil de instalar y configurar en la mayoría de las plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares).

- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de “convenir en vez de configurar”, en la que el desarrollador sólo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y puede ser adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

Symfony está desarrollado completamente con PHP 5, enfocado al desarrollo de aplicaciones en el mismo lenguaje de programación y está basado en un patrón clásico del diseño web conocido como arquitectura Modelo Vista Controlador (MVC).

### 1.7 Lenguaje de programación

**PHP:** (acrónimo de "PHP: Hypertext Preprocessor") es un lenguaje de "código abierto" interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor, utilizado para la generación de páginas Web dinámicas. Lo que distingue a PHP de la tecnología Javascript, la cual se ejecuta en la máquina cliente, es que el código PHP es ejecutado en el servidor. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas de sí mismo. Entre las habilidades de PHP se incluyen: creación de imágenes, archivos PDF y películas Flash. También puede presentar otros resultados, como XHTM y archivos XML. PHP puede auto generar estos archivos y almacenarlos en el sistema de archivos en vez de presentarlos en la pantalla. PHP puede además procesar la información de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies. Puede ser utilizado en cualquiera de los principales sistemas operativos del mercado, soporta la mayoría de los servidores Web, cuenta con soporte para comunicarse con

otros servicios usando protocolos de comunicación tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) etc. y soporta una gran cantidad de bases de datos.

Entre las principales ventajas de usar PHP se destacan que es extremadamente simple para un programador principiante, y a su vez, ofrece muchas características avanzadas para los programadores profesionales y el hecho de que se distribuye de forma gratuita bajo una licencia abierta.

### 1.8 Sistema gestor de base de datos

**PostgreSQL:** Es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) de código abierto considerado como el más avanzado del mundo. Posee muchas características que tradicionalmente sólo se podían ver en productos comerciales de alto calibre. Fue el pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, incluidos más tarde en otros sistemas de gestión comerciales. PostgreSQL es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. PostgreSQL puede ejecutarse sobre la mayoría de los sistemas operativos conocidos como Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), y Windows. Provee soporte para una amplia variedad de tipos nativos, y ofrece la posibilidad a los usuarios de crear sus propios tipos de datos. Otra de sus características favorables es que mediante un sistema denominado MVCC, acceso concurrente multiversión (*Multi-Version Concurrency Control*) permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.

## 1. 9 Conclusiones

En este capítulo se describieron algunos de los sistemas de gestión académica que son utilizados en el mundo y en nuestro país. Se realizó una valoración basada en las características de los mismos y en las deseadas para el nuevo módulo, estableciendo la necesidad de la realización de este trabajo puesto que ninguno de los sistemas estudiados responden a las necesidades existentes en cuanto a la gestión académica. Se fundamentó el uso de la metodología de desarrollo RUP y la herramienta CASE Visual Paradigm para la realización del trabajo, y se mencionan las características principales del framework, lenguaje de programación y sistema gestor de base de datos propuesto para el desarrollo del sistema.

# CAPÍTULO 2: DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA

## 2.1 Introducción

En el presente capítulo se modelará el negocio identificado, detallando cada uno de los procesos involucrados para dar solución a los problemas existentes. Se describen las reglas del negocio a tener en cuenta para el desarrollo del módulo así como las características que debe tener el mismo, expresado en los requisitos funcionales y no funcionales, a partir de los cuales se identifican los casos de uso del sistema y sus descripciones.

## 2.2 Principales reglas del negocio a considerar

Con el objetivo de garantizar las restricciones que existen en el negocio se deben tener en cuenta, a la hora de automatizar los procesos de gestión de los profesores las siguientes reglas:

- Un profesor debe recibir posteriormente a la contrata su ubicación.
- El Jefe de Departamento es el encargado de asignarle al profesor la asignatura y los grupos en los que impartirá clases.
- La ubicación de los profesores ocurre al comenzar cada momento del curso escolar.
- Un profesor puede solicitar su baja solo al finalizar el curso escolar.
- Cuando un profesor debe ausentarse un período de tiempo de la universidad por encontrarse cumpliendo otra tarea, el jefe de departamento debe asignar sus grupos a otro profesor de la asignatura o del departamento.

## 2.3 Actores del negocio

Un actor del negocio es cualquier persona o sistema externo que interactúa con el negocio y se beneficia de sus resultados.

*Tabla 2.1 Actores del Negocio.*

| Actor | Descripción |
|---|---|
| Profesor | Es quien inicia las acciones en el negocio. Puede solicitar su ubicación, su baja o ausentase del centro de estudios. |

## 2.4 Trabajadores del negocio

Un trabajador del negocio es cualquier persona o sistema externo que actúa con el negocio, realiza una o varias actividades y no se beneficia del mismo, manipulan entidades del negocio y pueden interactuar con otros trabajadores del negocio. El diagrama de objetos del negocio que se modela en el Anexo 1 muestra las relaciones que se expresan entre los trabajadores y las entidades del negocio.

*Tabla 2.2 Trabajadores del Negocio*

| Trabajador | Descripción |
|---|---|
| Decano. | Es aquel que está encargado de ubicar al profesor. Además es el encargado de analizar y aprobar las solicitudes de baja de los profesores. |
| Jefe de Departamento. | Es aquel que está encargado de asignar a un profesor la asignatura y grupo en el que impartirá clases. |

## 2.5 Diagrama de casos de uso del negocio

El modelo de casos de uso del negocio describe los procesos del negocio en términos de casos de uso del negocio y actores del negocio, que se corresponden con los procesos del negocio y los clientes de éste respectivamente. Este modelo se describe mediante diagramas de casos de uso que muestran la relación entre los casos de uso y los actores del negocio. En el Anexo 2 se modela el diagrama de actividades para cada caso de uso.

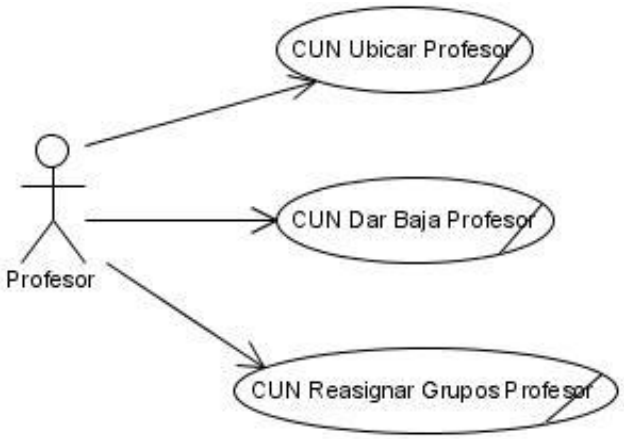


*Figura 1 Diagrama de CU del negocio*

## 2.6 Descripciones textuales de los casos de uso del negocio

Los casos de uso del negocio describen las actividades que se llevan a cabo en el negocio a partir de una secuencia de eventos donde se especifica la interacción de los actores del negocio con los trabajadores del mismo.

*Tabla 2.3 Descripción de CU Ubicar Profesor*

| **Caso de uso del negocio** | **CUN_Ubicar_Profesor** |
|---|---|
| Actores | Profesor |
| Resumen | El caso de uso se inicia cuando el profesor solicita su ubicación una vez que ha sido contratado por el centro. El caso de uso termina cuando se ha ubicado al profesor y se le ha asignado la asignatura que deberá impartir así como los grupos que debe atender. |
| Casos de uso asociados | |
| **Acción del actor** | **Respuesta del proceso de negocio** |
| 1. El profesor se presenta en la entidad en la que fue contratado y solicita su ubicación. | 2. El decano realiza una entrevista al profesor con el objetivo de conocer su especialidad, años de graduado, experiencia laboral y motivaciones y le solicita los documentos correspondientes. |

| | |
|---|---|
| 3. El profesor presenta su título de graduado y el currículo laboral. | 4. El decano verifica el título de graduado del profesor y su currículo laboral.<br><br>5. El decano consulta el listado de profesores por departamento y ubica al profesor en un departamento atendiendo a los resultados de la entrevista previamente realizada y a las necesidades de la facultad.<br><br>6. El decano actualiza el listado de profesores por departamento, informa al profesor en el departamento donde ha sido ubicado, le orienta ver al jefe de departamento. |
| 7. El profesor se presenta en el Departamento donde fue ubicado y solicita su ubicación. | 8. El Jefe de Departamento consulta el listado de profesores por asignatura y grupos y le asigna una de ellas al profesor así como los grupos a los que impartirá clases atendiendo a las necesidades del departamento.<br><br>9. El Jefe de Departamento actualiza el listado de profesores por asignatura y grupos.<br><br>10. Informa al profesor. |
| Otras secciones: | - |

*Tabla 2.4 Descripción CU Dar Baja Profesor*

| **Caso de uso del negocio** | **CUN_Dar_Baja_Profesor** |
|---|---|
| Actores | Profesor |
| Resumen | El caso de uso comienza cuando un profesor al finalizar el curso solicita su baja del centro de estudios. El caso de uso concluye cuando se aprueba o no la baja por parte de los directivos. |
| Casos de uso asociados | |
| **Acción del actor** | **Respuesta del proceso de negocio** |

| | |
|---|---|
| 1. Solicita su baja del centro. | 2. El Decano verifica si el profesor se encuentra cumpliendo su período de adiestramiento o servicio social en el listado de profesores por departamento. Ver flujo alterno 1.<br><br>3. El Decano aprueba la baja del profesor y le informa al mismo.<br><br>4. Elimina al profesor del listado de profesores por departamentos.<br><br>5. El jefe de departamento elimina al profesor en el listado de profesores por asignaturas y grupos. |
| Flujo alterno 1: | |
| | El Decano analiza si puede aprobar la baja. Ver flujo alterno 2.<br><br>No aprueba la baja del centro.<br><br>Se informa al profesor. |
| Flujo alterno 2: | |
| | El Decano coordina para que el profesor culmine su período de adiestramiento o servicio social en otro centro laboral.<br><br>Se informa al profesor dónde debe concluir su período de adiestramiento o servicio social.<br><br>Volver a la acción 4 del flujo normal de eventos. |
| Otras secciones: | - |

*Tabla 2.5 Descripción CU Reasignar Grupos Profesor*

| **Caso de uso del negocio** | **CUN_Reasignar_Grupos_Profesor** |
|---|---|
| Actores | Profesor |
| Resumen | El caso de uso comienza cuando un profesor debe ausentarse del centro en el período docente por algún motivo. El caso de uso termina cuando son asignados sus grupos a otro profesor del departamento. |
| Casos de uso asociados | |
| **Acción del actor** | **Respuesta del proceso de negocio** |
| 1. El profesor debe ausentarse del centro en el período docente y se le comunica al jefe de departamento. | 2. El Jefe de Departamento busca la asignatura que imparte el profesor y los grupos que atiende según el listado de profesores por asignatura y grupos.<br><br>3. El Jefe de Departamento consulta el listado de profesores por asignatura y grupos para analizar la carga docente de los profesores que imparten la misma asignatura.<br><br>4. Se decide el profesor de la asignatura que puede asumir los grupos libres.<br><br>5. Si no hay un profesor de la asignatura que pueda asumir los grupos. Ver flujo alterno 1.<br><br>6. Se le asignan los grupos libres al profesor.<br><br>7. Se registra en el listado de profesores por asignatura y grupos.<br><br>8. Se informa al profesor. |
| Flujo Alterno 1: | El Jefe de Departamento consulta el listado de |

| | |
|---|---|
| | profesores por asignatura para analizar la carga docente de todos los profesores del departamento.<br><br>El Jefe de Departamento define el profesor del departamento que va a impartir la asignatura en los grupos libres.<br><br>Se le asigna la asignatura y los grupos libres al profesor.<br><br>Volver a la acción 7 del flujo normal de eventos. |
| Otras secciones: | - |

## 2.7 Técnicas de recopilación de requisitos

Desde el inicio del desarrollo de sistemas ha estado presente el gran problema de la identificación de los requisitos, debido a que no es un proceso que pueda ser determinado matemáticamente, en el cual los datos son extraídos de las personas interesadas, lo que hace que estos puedan variar en dependencia de la persona a la que se consulte. Todo esto hace que la ingeniería de requisitos se considere como una etapa clave en el desarrollo de software. Es el proceso de definir los servicios que el cliente requiere del sistema y las restricciones bajo las cuales éste opera y es desarrollado. Su importancia es considerada esencial, pues los errores más comunes y más costosos de reparar, así como los que más tiempo consumen se deben a una inadecuada ingeniería de requisitos. Mediante el tratamiento de requisitos se especifican los servicios que debe proporcionar el sistema, consiste en un proceso iterativo y cooperativo de análisis del problema. Todo este proceso de recopilar información y definir las necesidades del sistema son procesos complejos, pues hay que identificar los requisitos que el sistema debe cumplir para satisfacer las necesidades de los usuarios finales y de los clientes, además para que el sistema sea considerado un software de calidad.(9)

La captura de requisitos es una parte importante en el desarrollo del software, ya que permite entender el negocio así como identificar sus aspectos positivos y negativos. Para lograr un buen resultado en esta etapa del desarrollo es necesario mantener una buena comunicación con los clientes y los usuarios de manera que se puedan conocer sus necesidades y las expectativas que tienen sobre el sistema a desarrollar.

Para la realización del levantamiento de requisitos del nuevo Módulo Profesor se utilizaron diferentes técnicas, como la Tormenta de Ideas, a través de la misma los analistas del proyecto, en conjunto con el equipo de desarrollo, se reunieron de forma informal y emitieron ideas teniendo como base el estudio de las funcionalidades del sistema existente y las problemáticas presentadas alrededor del mismo, de esta manera se tuvieron varias vistas del problema y la propuesta de algunos requisitos aún difusos.

Un vez concluida esta etapa se continuó con la identificación y captura requisitos de forma más detallada, para lo que se utilizó la técnica conocida como Entrevista. Se prepararon varias entrevistas con el objetivo de constatar las opiniones de los involucrados directamente en los procesos del negocio. En estos espacios el cliente fue el principal protagonista y los analistas guiaron la conversación en torno al problema, este método generó una mayor implicación de los clientes y usuarios. Esta técnica permitió detallar los requisitos e identificar otros que no se tuvieron en cuenta anteriormente.

Posteriormente se organizaron reuniones en las cuales, a través de una presentación visual, se les mostró a los clientes y usuarios, de forma dinámica y en un lenguaje natural entendible por todos, un resumen con ideas de la estructura del sistema a desarrollar. Estas reuniones permitieron llegar a un acuerdo común sobre las funcionalidades que el módulo debe cumplir validando las mismas a través de un documento firmado por los clientes como constancia.

Otra de las técnicas empleadas fue la de los Casos de Uso, ésta se utilizó principalmente para la especificación de los requisitos funcionales del sistema. Se realizó una descripción textual de los casos de uso que responden a los requisitos funcionales, utilizando las plantillas que propone RUP, describiendo las interacciones en un lenguaje natural, fácilmente comprensible por los clientes y usuarios.

### 2.7.1 Requerimientos funcionales

Los requisitos funcionales manifiestan las capacidades o condiciones que el sistema debe cumplir, son independientes de las propiedades o cualidades que debe tener el software y permiten establecer un acuerdo con los clientes que debe ser mantenido a lo largo del desarrollo del sistema.

**R1 Adicionar profesor**

1. Mostrar la interfaz visual con las opciones posibles para la ubicación del profesor.
2. Obtener los datos del profesor de forma manual o automática utilizando la plantilla Datos Personales del Profesor, definida con anterioridad.
    a. Obtener los datos de un profesor de forma automática.
        i. Mostrar la interfaz visual para la búsqueda de un profesor.
        ii. Permitir buscar a un profesor por diferentes criterios de búsqueda: Nombre, Apellidos, Carné de Identidad.
        iii. Listar todos los profesores que cumplan con el/los criterios de búsqueda mostrando Nombre, Apellidos, Foto.
            1. Mostrar un mensaje en caso de que no se encuentre ningún profesor que cumpla con el/los parámetros de búsqueda y permitir introducirlos nuevamente.
        iv. Seleccionar al profesor deseado.
        v. Mostrar los datos referentes al profesor seleccionado: Datos definidos en la plantilla Datos Personales del Profesor, Estructuras a la que pertenece, Tipo de Profesor, Categoría Docente, Categoría Científica.

vi. Chequear que el profesor a adicionar no esté registrado en la estructura.
    1. En caso de que el profesor esté registrado, mostrar un mensaje de error.
vii. Chequear que el profesor a adicionar no esté registrado en otra estructura.
    1. En caso de que el profesor esté registrado en otra estructura, permitir modificar el campo Tipo de Profesor.
    2. En caso de que el profesor no esté registrado en otra estructura, permitir seleccionar: Tipo de Profesor, Categoría Docente y Categoría Científica.

b. Permitir introducir los datos del profesor de forma manual.
    i. Mostrar según la plantilla Datos Personales del Profesor, definida con anterioridad, los campos referentes a los datos que debe introducir.
    ii. Validar los datos del profesor definidos en la plantilla Datos Personales del Profesor.
        1. En caso de que los datos introducidos estén mal, mostrar un mensaje y permitir introducirlos nuevamente
    iii. Chequear que el profesor a adicionar no esté registrado en el sistema.
        1. En caso de que el profesor esté registrado en el sistema, mostrar un mensaje y permitir obtener los datos de forma automática.
    iv. Permitir seleccionar los datos asociados al profesor: Tipo de Profesor, Categoría Docente, Categoría Científica.

3. Asignar el estado del profesor como Activo.
4. Almacenar los datos referentes del profesor definidos en la plantilla Datos Personales del Profesor, Tipo de Profesor, Categoría Docente, Categoría Científica y el estado asignado.
5. Registrar incidencia.

**R2 Modificar profesor**

1. Mostrar la interfaz visual para la búsqueda de un profesor.
2. Permitir buscar a un profesor por diferentes criterios de búsqueda: Nombre, Apellidos, Carné de Identidad.
3. Listar todos los profesores que cumplan con el/los criterios de búsqueda mostrando Nombre, Apellidos, Foto.
    a. Mostrar un mensaje en caso de que no se encuentre ningún profesor que cumpla con el/los parámetros de búsqueda y permitir introducirlos nuevamente.
4. Seleccionar al profesor deseado.
5. Mostrar los datos referentes al profesor seleccionado: Datos definidos en la plantilla Datos Personales del Profesor, Estructuras a la que pertenece, Tipo de Profesor, Categoría Docente, Categoría Científica.
6. Permitir modificar los datos del profesor: Datos definidos en la plantilla Datos Personales del Profesor, Tipo de Profesor, Categoría Docente, Categoría Científica.
7. Validar los datos del profesor definidos en la plantilla Datos Personales del Profesor.
    a. Si existe error en los datos del profesor definidos en la plantilla Datos Personales del Profesor, mostrar un mensaje y permitir introducirlos nuevamente.
8. Almacenar los datos del profesor: Datos definidos en la plantilla Datos Personales del Profesor, Tipo de Profesor, Categoría Docente, Categoría Científica.
9. Registrar Incidencia.

**R3 Eliminar profesor**

1. Mostrar la interfaz visual para la búsqueda de un profesor.
2. Permitir buscar a un profesor por diferentes criterios de búsqueda: Nombre, Apellidos, Carné de Identidad.

3. Listar todos los profesores que cumplan con el/los criterios de búsqueda mostrando Nombre, Apellidos, Foto.
    a. Mostrar un mensaje en caso de que no se encuentre ningún profesor que cumpla con el/los parámetros de búsqueda y permitir introducirlos nuevamente.
4. Seleccionar al profesor deseado.
5. Mostrar los datos referentes al profesor seleccionado: Datos definidos en la plantilla Datos Personales del Profesor, Estructuras a la que pertenece, Tipo de Profesor, Categoría Docente, Categoría Científica.
6. Chequear si el profesor a eliminar tiene asignados grupos.
    a. Permitir, en caso de que el profesor tenga asignados grupos, eliminar estas asignaciones.
    b. En caso de eliminar alguna de las asignaciones, mostrar un mensaje indicando que el grupo en el que se eliminó la asignación, no cuenta con profesor en la asignatura correspondiente a la eliminada.
7. Si el profesor a eliminar tiene asignados asignaturas y grupos y no fueron eliminadas estas asignaciones, mostrar un mensaje indicando que no puede ser eliminado el profesor.
8. Modificar el estado del profesor a “Inactivo”.
9. Registrar Incidencia.

**R4 Buscar profesor**

1. Mostrar la interfaz visual para la búsqueda de un profesor.
2. Permitir buscar a un profesor por diferentes criterios de búsqueda: Nombre, Apellidos, Carné de Identidad.
3. Listar todos los profesores que cumplan con el/los criterios de búsqueda mostrando Nombre, Apellidos, Foto.
    a. Mostrar un mensaje en caso de que no se encuentre ningún profesor que cumpla con el/los parámetros de búsqueda y permitir introducirlos nuevamente.
4. Seleccionar al profesor deseado.

5. Mostrar los datos referentes al profesor seleccionado: Datos definidos en la plantilla Datos Personales del Profesor, Estructuras a la que pertenece, Tipo de Profesor, Categoría Docente, Categoría Científica.
6. Registrar incidencia.

**R5 Adicionar grupos al profesor**

1. Mostrar la interfaz visual para la búsqueda de un profesor.
2. Permitir buscar a un profesor por diferentes criterios de búsqueda: Nombre, Apellidos, Carné de Identidad.
3. Listar todos los profesores que cumplan con el/los criterios de búsqueda mostrando Nombre, Apellidos, Foto.
    a. Mostrar un mensaje en caso de que no se encuentre ningún profesor que cumpla con el/los parámetros de búsqueda y permitir introducirlos nuevamente.
4. Seleccionar al profesor deseado.
5. Mostrar los datos referentes al profesor seleccionado: Datos definidos en la plantilla Datos Personales del Profesor, Estructuras a la que pertenece, Tipo de Profesor, Categoría Docente, Categoría Científica.
6. Permitir realizar las asignaciones de grupos a un profesor.
    a. Listar las asignaciones que ya tiene el profesor mostrando, asignatura y grupo.
    b. Permitir realizar nuevas asignaciones al profesor.
        i. Especificar para cada asignación: Disciplina, Plan de Estudios, Nivel, Momento, Tipo de asignatura, Asignatura, Grupo Docente.
    c. Chequear que el profesor no tenga realizada esa asignación.
        i. En caso de que el profesor ya tenga la asignación, mostrar un mensaje y permitir realizar una nueva asignación.
    d. Almacenar los datos del profesor: Disciplina, Plan de Estudios, Nivel, Momento, Tipo de asignatura, Grupo Docente.
7. Registrar Incidencia.

**R6 Eliminar grupos al profesor**

1. Mostrar la interfaz visual para la búsqueda de un profesor.
2. Permitir buscar a un profesor por diferentes criterios de búsqueda: Nombre, Apellidos, Carné de Identidad.
3. Listar todos los profesores que cumplan con el/los criterios de búsqueda mostrando Nombre, Apellidos, Foto.
    a. Mostrar un mensaje en caso de que no se encuentre ningún profesor que cumpla con el/los parámetros de búsqueda y permitir introducirlos nuevamente.
4. Seleccionar al profesor deseado.
5. Mostrar los datos referentes al profesor seleccionado: Datos definidos en la plantilla Datos Personales del Profesor, Estructuras a la que pertenece, Tipo de Profesor, Categoría Docente, Categoría Científica.
6. Permitir eliminar la asignación de los grupos que atiende un profesor.
    a. Eliminar la asignación de la asignatura que imparte en el grupo.
7. Mostrar un mensaje indicando que el grupo en el que se eliminó la asignación, no cuenta con profesor en la asignatura correspondiente a la eliminada.
8. Actualizar los datos del profesor: Asignaturas y Grupos.
9. Registrar Incidencia.

**R7 Crear tipo de profesor**

1. Mostrar el formulario para la creación de un tipo de profesor.
2. Introducir los datos referentes al tipo de profesor a crear: Nombre, Descripción.
3. Chequear que los datos estén correctos.
    a. Si los datos están incorrectos, mostrar un mensaje de error y permitir volver a introducirlos.
4. Chequear que no exista un tipo de profesor con el mismo Nombre del que se desea crear.
    a. Si existe un tipo de profesor con el mismo nombre del que se desea crear, mostrar un mensaje de error y permitir introducir los datos nuevamente.
5. Almacenar la información del tipo de profesor creado: Nombre, Descripción

6. Registrar incidencia.

**R8 Actualizar tipo de profesor**

1. Listar todos los tipos de profesores definidos anteriormente mostrando el nombre.
2. Seleccionar el tipo de profesor a modificar.
3. Mostrar los datos del tipo de profesor seleccionado: Nombre, Descripción.
4. Permitir modificar el nombre y /o la descripción del tipo de profesor seleccionado.
5. Chequear que los datos estén correctos.
    a. Si los datos están incorrectos, mostrar un mensaje de error y permitir volver a introducirlos.
6. Si se modifica el nombre:
    a. Chequear que no exista un tipo de profesor con el nuevo nombre definido.
    b. Si existe un tipo de profesor con el mismo nombre del que se desea crear, mostrar un mensaje de error y permitir introducir los datos nuevamente.
7. Modificar el campo nombre y/o la descripción del tipo de profesor a los profesores que tengan asignado el tipo de profesor modificado.
8. Almacenar la información modificada: Nombre, Descripción.
9. Registrar incidencia.

**R9 Eliminar tipo de profesor**

1. Listar todos los tipos de profesores definidos anteriormente mostrando el nombre.
2. Seleccionar el tipo de profesor a eliminar.
3. Mostrar los datos del tipo de profesor seleccionado: Nombre, Descripción.
4. Comprobar que no existan profesores que tengan asignado el tipo de profesor a eliminar.
    a. Si existen profesores que tengan asignado el tipo de profesor a eliminar, mostrar un mensaje de error indicando que no puede ser eliminado.

5. Eliminar el tipo de profesor seleccionado.
6. Registrar Incidencia.

**R10 Crear categoría docente**

1. Mostrar el formulario para la creación de una categoría docente.
2. Introducir los datos referentes a la categoría docente a crear: Nombre, Descripción.
3. Chequear que los datos estén correctos.
    a. Si los datos están incorrectos, mostrar un mensaje de error y permitir volver a introducirlos.
4. Chequear que no exista una categoría docente con el mismo Nombre de la que se desea crear.
    a. Si existe una categoría docente con el mismo nombre de la que se desea crear, mostrar un mensaje de error y permitir introducir los datos nuevamente.
5. Almacenar la información de la categoría docente creada: Nombre, Descripción.
6. Registrar incidencia.

**R11 Actualizar categoría docente**

1. Listar todas las categorías docentes definidas anteriormente mostrando el nombre.
2. Seleccionar la categoría docente a modificar.
3. Mostrar los datos de la categoría docente seleccionada: Nombre, Descripción.
4. Permitir modificar el nombre y /o la descripción de la categoría docente seleccionada.
5. Chequear que los datos estén correctos.
    a. Si los datos están incorrectos, mostrar un mensaje de error y permitir volver a introducirlos.
6. Si se modifica el nombre:

a. Chequear que no exista una categoría docente con el nuevo nombre definido.
b. Si existe una categoría docente con el mismo nombre de la que se desea crear, mostrar un mensaje de error y permitir introducir los datos nuevamente.

7. Modificar el dato nombre y/o descripción de la categoría docente a los profesores que tengan asignada la categoría docente modificada.
8. Almacenar la información modificada: Nombre, Descripción.
9. Registrar incidencia.

**R12 Eliminar categoría docente**

1. Listar todas las categorías docentes definidas anteriormente mostrando el nombre.
2. Seleccionar la categoría docente a eliminar.
3. Mostrar los datos de la categoría docente seleccionada: Nombre, Descripción.
4. Comprobar que no existan profesores que tengan asignada la categoría docente a eliminar.
    a. Si existen profesores que tengan asignada la categoría docente a eliminar, mostrar un mensaje de error indicando que no puede ser eliminada.
5. Eliminar la categoría docente seleccionada.
6. Registrar incidencia.

**R13 Crear categoría científica.**

1. Mostrar el formulario para la creación de una categoría científica.
2. Introducir los datos referentes a la categoría científica a crear: Nombre, Descripción.
3. Chequear que los datos estén correctos.
    a. Si los datos están incorrectos, mostrar un mensaje de error y permitir volver a introducirlos.

4. Chequear que no exista una categoría científica con el mismo Nombre de la que se desea crear.
    a. Si existe una categoría científica con el mismo nombre de la que se desea crear, mostrar un mensaje de error y permitir introducir los datos nuevamente.
5. Almacenar la información de la categoría científica creada: Nombre, Descripción.
6. Registrar incidencia.

**R14 Actualizar categoría científica**

1. Listar todas las categorías científicas definidas anteriormente mostrando el nombre.
2. Seleccionar la categoría científica a modificar.
3. Mostrar los datos de la categoría científica seleccionada: Nombre, Descripción.
4. Permitir modificar el nombre y /o la descripción de la categoría científica seleccionada.
5. Chequear que los datos estén correctos.
    a. Si los datos están incorrectos, mostrar un mensaje de error y permitir volver a introducirlos.
6. Si se modifica el nombre:
    a. Chequear que no exista una categoría científica con el nuevo nombre definido.
    b. Si existe una categoría científica con el mismo nombre de la que se desea crear, mostrar un mensaje de error y permitir introducir los datos nuevamente.
7. Modificar el dato nombre y/o descripción de la categoría científica a los profesores que tengan asignada la categoría científica modificada.
8. Almacenar la información modificada: Nombre, Descripción.
9. Registrar incidencia.

**R15 Eliminar categoría científica**

1. Listar todas las categorías científicas definidas anteriormente mostrando el nombre.
2. Seleccionar la categoría científica a eliminar.
3. Mostrar los datos de la categoría científica seleccionada: Nombre, Descripción.
4. Comprobar que no existan profesores que tengan asignada la categoría científica a eliminar.
    a. Si existen profesores que tengan asignada la categoría científica a eliminar, mostrar un mensaje de error indicando que no puede ser eliminada.
5. Eliminar la categoría científica seleccionada.
6. Registrar incidencia.

**R16 Consultar información**

1. Mostrar todas las opciones de reportes que se pueden consultar.
2. Permitir seleccionar el reporte que se desea consultar.
    a. Si se desea consultar el listado de profesores por estructura.
        i. Listar los profesores de la estructura, mostrando de cada profesor: Nombre, Apellidos, Foto.
            1. Mostrar un mensaje en caso de que no existan profesores registrados en la estructura.
        ii. Registrar incidencia.
    b. Si se desea consultar el listado de profesores por disciplina.
        i. Mostrar el listado de disciplinas.
        ii. Permitir seleccionar la disciplina deseada.
        iii. Listar los profesores de la disciplina mostrando para cada profesor: Nombre, Apellidos, Foto, Asignatura de la disciplina que imparte.
            1. Mostrar un mensaje en caso de que no existan profesores de la disciplina.
        iv. Registrar incidencia.

c. Si se desea consultar el listado de profesores por asignatura.
    i. Mostrar el listado de asignaturas.
    ii. Permitir seleccionar la asignatura deseada.
    iii. Listar los profesores de la asignatura mostrando para cada profesor: Nombre, Apellidos, Foto, Grupo donde imparte la asignatura.
        1. Mostrar un mensaje en caso de que no existan profesores de la asignatura.
    iv. Registrar incidencia.
d. Si se desea consultar el listado de profesores por tipo de profesor.
    i. Mostrar el listado de tipos de profesores definidos.
    ii. Permitir seleccionar el tipo de profesor deseado.
    iii. Listar los profesores que tienen asignado el tipo de profesor seleccionado mostrando: Nombre, Apellidos, Foto.
        1. Mostrar un mensaje en caso de que no existan profesores con ese tipo de profesor asignado.
    iv. Registrar incidencia.

### 2.7.2 Requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Son las características que hacen al producto atractivo, usable, rápido y confiable.

**Usabilidad**

1. Para los usuarios normales se requerirá un tiempo de entrenamiento de 15 días.
2. Para los usuarios avanzados se requerirá un tiempo de entrenamiento de 7 días.
3. Estándares de Usabilidad.
    a. Generales
        i. Hacer uso de gráficos e imágenes claras y nítidas.

ii. Limitar el número de enlaces en una página.
iii. Evitar el uso de pequeños botones y enlaces con texto minúsculo.
iv. Respetar el espacio entre los enlaces y botones.
v. Evitar el uso de los menús en cascada.
vi. Proporcionar a los formularios un título que exprese claramente su función.

b. Texto
   i. Utilizar una nomenclatura clara y familiar.
   ii. Evitar el uso de preguntas complejas.
   iii. Redactar siempre las opciones de forma afirmativa.

c. Organización
   i. Organizar los campos en una sola columna de datos.
   ii. Agrupar, siempre que sea posible, los campos obligatorios al comienzo del formulario.
   iii. Evitar fragmentar la petición de información.
   iv. Proporcionar un diseño ordenado, alineando verticalmente todas las etiquetas y todos los campos entre si.

d. Tipos de campos
   i. Hacer corresponder el tamaño visible de los campos de texto con la longitud del contenido que ha de introducir el usuario.
   ii. Seleccionar siempre una opción por defecto, si se utilizan combos o radio buttons**,** asegurándose de que sea la más probable.

e. Funcionamiento
   i. Asegurar que la tecla "Intro" realiza la acción principal.
   ii. Evitar que el usuario pueda impacientarse y enviar dos veces el formulario.

f. Ayudas
   i. Identificar claramente los campos obligatorios y los opcionales mediante el literal (Obligatorio) u (Opcional).

g. Botones
   i. Evitar tener un botón "Cancelar" cuya función sea en realidad volver a la página anterior.

ii. Dar un nombre adecuado a los botones del formulario, relacionado con su acción y no de carácter general.

h. Errores
    i. Mantener la información en los campos no erróneos cuando se produzca un error, de forma tal que se mantenga la información introducida.

i. FeedBack
    i. Informar el resultado de una acción al usuario cuando se ejecute.
    ii. Incluir cláusulas de protección de datos cuando sea necesario.

j. Accesibilidad
    i. Permitir, a través del tabulador, acceder a todos los campos en el mismo orden que el visual.

k. Formularios extensos
    i. Dividir en páginas los formularios muy extensos.
    ii. Permitir al usuario volver a los pasos anteriores.
    iii. Evitar la utilización de pestañas para crear formularios de varias páginas.

**Fiabilidad**

1. El sistema debe estar disponible las 24 horas del día y los 7 días de la semana.
2. Hacer transparente para el usuario el mantenimiento del sistema.
3. El tiempo medio permitido para que el sistema quede fuera de operación luego de haber fallado debe ser como máximo 7 días.
4. El tiempo medio entre fallos debe ser como máximo un año.
5. Las salidas del sistema deben ser de una alta precisión y exactitud de acuerdo a las necesidades del usuario.
6. Proteger la aplicación de accesos no autorizados.
7. Permitir el acceso de los usuarios autenticados a las áreas definidas para su rol.
8. Registrar todas las incidencias de los usuarios en el sistema.
9. Proteger la información generada por el sistema
10. Los errores oscilaran entre 50 y 250 errores/MLC

11. Clasificar los errores en menores, significativos y críticos.
    a. Menores: No resultan peligrosos porque no causa daños significativos al sistema.
    b. Significativos: Pueden causar daños mayores a la aplicación, pero si se actúa de manera correctiva inmediata puede lograrse la supervivencia del sistema.
    c. Crítico: Pueden causar daños fatales de información, o la pérdida del sistema completo.

**Eficiencia**

1. El tiempo de respuesta por transacción debe ser entre 2 segundos y 5 segundos.
2. Se desea un sistema eficiente con gran nivel de precisión, con tiempo de procesamiento de información y tiempo de respuestas rápidos, y que mantenga siempre la consistencia de los datos.
3. El sistema deberá operar con grandes volúmenes de información y con un número de clientes mayor a 10 mil, por lo que se necesita de alta capacidad de concurrencia del servidor de datos y del servidor web donde se encuentre alojada la aplicación.
4. Utilización de recursos.

   **Para Desarrollo:**

   - Intel Pentium 4 o superior
   - CPU 3GHZ o superior
   - 512 MB RAM o superior
   - 20 GB HDD o superior
   - Tarjeta de red con velocidad mínima de 100 Mbs

   **Para Explotación:**

   Clientes:

   - Pentium 3 o superior
   - CPU 133 MHZ o superior
   - 128 RAM mínimo 512 RAM recomendada o superior.

- Tarjeta de red con velocidad mínima de 100 Mbs

Servidor Web y de Base de Datos:

- CPU: intel Core 2 Duo 2.0 GHZ o superior
- RAM: 6 GB
- 250 GB HDD

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

**Soporte**

1. Realizar distintas pruebas al software una vez concluido para comprobar su funcionalidad.
2. Ofrecer los servicios de instalación y configuración de la aplicación una vez terminado el software.
3. Prestar servicios de mantenimiento del software.
4. Para el servidor de aplicaciones se requiere que esté instalado un intérprete de ficheros PHP y con las últimas actualizaciones del lenguaje.
5. Para el servidor de base de datos se requiere que esté instalado un gestor de base de datos que soporte grandes volúmenes de datos y velocidad de procesamiento.

**Restricciones de diseño**

- Visual Paradigm para el análisis y diseño del módulo, utilizando RUP como metodología de desarrollo y UML como lenguaje de modelado.
- IDE de desarrollo Eclipse Europa 3.3.0.
- PHP 5 como lenguaje de programación.
- Symfony como framework de desarrollo.
- La arquitectura que se usara será Modelo Vista Controlador (MVC).

**Requisitos para la documentación de usuarios en línea y ayuda del sistema**

- Conformar una ayuda en línea para los usuarios finales donde tendrán permiso a ver la ayuda de acuerdo a las operaciones que pueden hacer en el sistema.

**Interfaz**

1. Las páginas no deben exceder los 100 Kb en las imágenes.
2. Mantener el mismo formato en todas las páginas.
3. Combinar correctamente los colores, tipo de letra y tamaño.
4. Los íconos deben estar en correspondencia con lo que representan.
5. Las páginas de la aplicación no cargarán con mucha información y contendrán sólo las imágenes necesarias.
6. Se utilizará como protocolo de comunicación http y https, este ultimo será utilizado para las transacciones que requieran mayor seguridad

**Interfaces hardware**

1. Soportar interfaces para impresoras.

**Interfaces de comunicación**

1. Brindar interfaces de comunicación para los sistemas que lo requieran garantizando que los cambios que se hagan en el sistema se actualicen en los terceros.

**Restricciones de Software:**

Cliente:

- Sistema Operativo con interfaz gráfica y conexión a red.
- Navegador Web.

Servidor:

- Ubuntu Server 7.10
- Symfony Framework 1.0.11
- PostgreSQL 8.1.2

## 2.8 Actores del sistema

Un actor del sistema es una persona, un equipo o un sistema automatizado, que sin ser parte del sistema interactúa con el mismo.

*Tabla 2.6 Actores del sistema*

| Actor | Descripción |
|---|---|
| Secretaria Docente. | Es quien está encargado de realizar todo el proceso de ubicación de un profesor, así como adicionarle o eliminarle los grupos en los que va a impartir clases. Además es la encargada de eliminar a un profesor y podrá tener acceso a toda la información del profesor. |
| Consultor. | Es cualquier directivo o la propia secretaria docente que puede tener acceso a los diferentes reportes del sistema referentes al profesor. |
| Administrador. | Es quien está encargado en el centro de estudios de definir los tipos de profesores y las categorías docentes con que van a contar. |

## 2.9 Diagrama de casos de uso del sistema

El diagrama de casos de uso del sistema muestra la interacción entre los actores del sistema y los procesos a automatizar representados a través de los casos de uso del sistema y que responden a los requisitos funcionales del mismo.

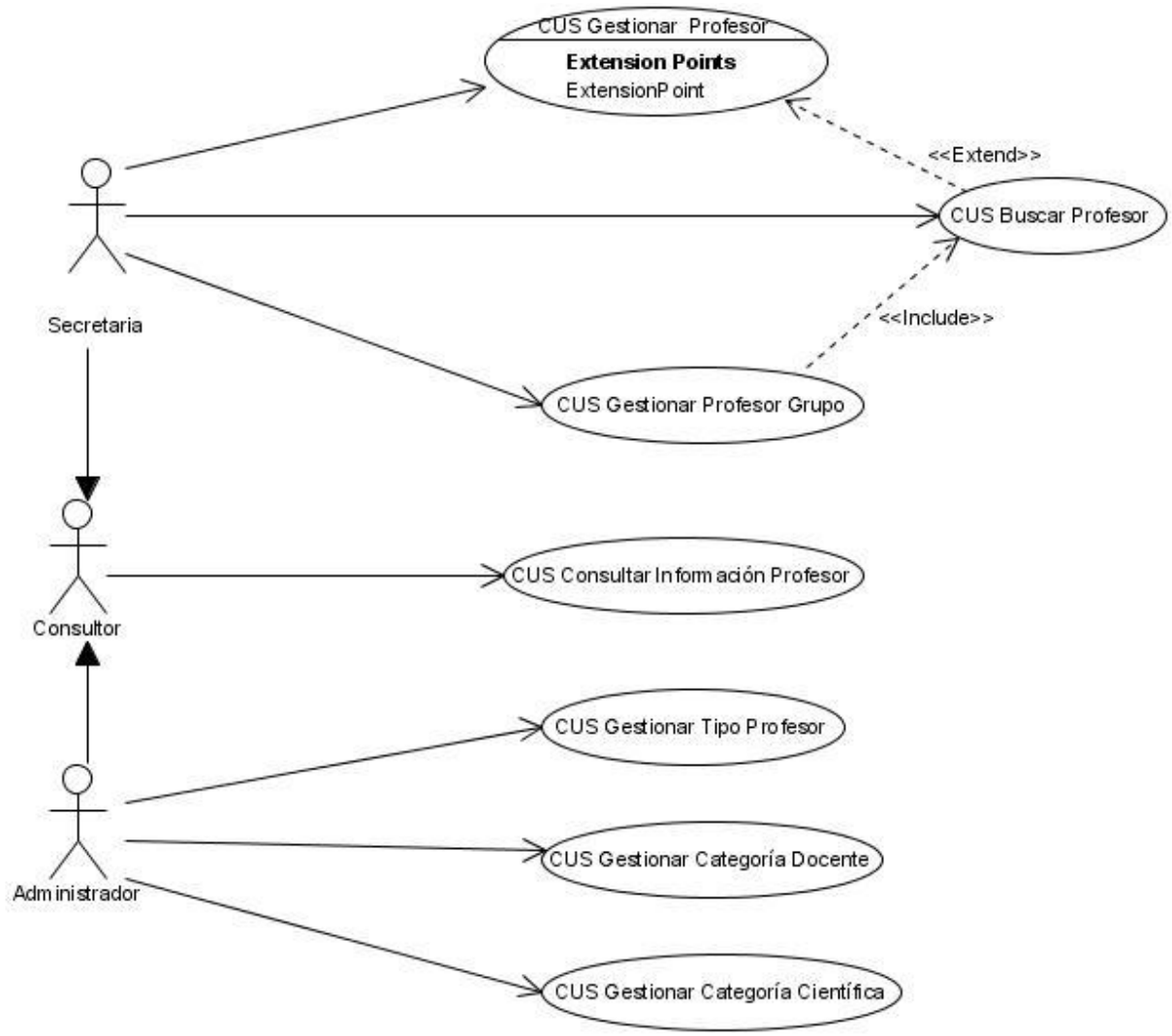


*Figura 2 Diagrama de casos de uso del sistema*

## 2.10 Descripciones abreviadas de los casos de uso del sistema.

La descripción de los casos de uso del sistema permite comprender el comportamiento del sistema desde el punto de vista del usuario. A continuación se muestran las descripciones abreviadas de los casos de uso del sistema, las descripciones detalladas pueden ser encontradas en el Anexo 3.

*Tabla 2.7 Resumen del CU Gestionar Profesor*

| | |
|---|---|
| **Caso de Uso:** | CUS_Gestionar_Profesor |
| **Actores:** | Secretaria Docente |
| **Resumen:** | El caso de uso permite adicionar, modificar y eliminar a un |

| | |
|---|---|
| | profesor en el sistema. |
| **Precondiciones:** | La secretaria debe tener los permisos necesarios para poder ejecutar esta acción.<br>Para eliminar a un profesor debe de estar registrado y su estado debe ser activo. |
| **Referencias** | R1, R2,R3 |
| **Prioridad** | Crítico. |

*Tabla 2.8 Resumen del CU Buscar Profesor*

| | |
|---|---|
| **Caso de Uso:** | CUS_Buscar_Profesor |
| **Actores:** | Secretaria Docente |
| **Resumen:** | El caso de uso permite buscar a un profesor en el sistema. |
| **Precondiciones:** | La secretaria debe estar registrada y tener los permisos necesarios para poder ejecutar esta acción.<br>El profesor tiene que estar registrado y su estado debe ser activo. |
| **Referencias** | R4 |
| **Prioridad** | Crítico. |

*Tabla 2.9 Resumen del CU Gestionar Profesor Grupo*

| | |
|---|---|
| **Caso de Uso:** | CUS_Gestionar_Profesor Grupo |
| **Actores:** | Secretaria Docente |
| **Resumen:** | El caso de uso permite adicionar y eliminar grupos docentes a un profesor. |
| **Precondiciones:** | La secretaria debe estar registrada y tener los permisos necesarios para poder ejecutar esta acción.<br>El profesor tiene que estar registrado y su estado debe ser activo. |
| **Referencias** | R5,R6 |
| **Prioridad** | Auxiliar |

*Tabla 2.10 Resumen del CU Gestionar Tipo Profesor*

| | |
|---|---|
| **Caso de Uso:** | CUS_Gestionar_Tipo_de_Profesor. |
| **Actores:** | Administrador |

| | |
|---|---|
| **Resumen:** | Este caso de uso permite adicionar, modificar y eliminar los tipos de profesores que pueden existir en el sistema. |
| **Precondiciones:** | El usuario debe estar registrado y tener los permisos necesarios para realizar la acción.<br>Para eliminar un tipo de profesor, debe de haber sido creado con anterioridad. |
| **Referencias** | R7, R8, R9 |
| **Prioridad** | Secundario. |

*Tabla 2.11 Resumen del CU Gestionar Categoría Docente*

| | |
|---|---|
| **Caso de Uso:** | CUS_Gestionar_Categoría_Docente. |
| **Actores:** | Administrador |
| **Resumen:** | Este caso de uso permite adicionar, modificar y eliminar los tipos de categoría docente que pueden existir en el sistema. |
| **Precondiciones:** | El usuario debe estar registrado y tener los permisos necesarios para realizar la acción.<br>Para eliminar una categoría docente, debe de haber sido creada con anterioridad. |
| **Referencias** | R10, R11, R12 |
| **Prioridad** | Secundario. |

*Tabla 2.12 Resumen del CU Gestionar Categoría Científica*

| | |
|---|---|
| **Caso de Uso:** | CUS_Gestionar_Categoría_Científica. |
| **Actores:** | Administrador |
| **Resumen:** | Este caso de uso permite adicionar, modificar y eliminar los tipos de categoría científica que pueden existir en el sistema. |
| **Precondiciones:** | El usuario debe estar registrado y tener los permisos necesarios para realizar la acción.<br>Para eliminar una categoría científica, debe de haber sido creada con anterioridad. |
| **Referencias** | R13, R14, R15 |
| **Prioridad** | Secundario. |

*Tabla 2.13 Resumen del CU Consultar información Profesor*

| | |
|---|---|
| **Caso de Uso:** | CUS_Consultar_Información_Profesor. |
| **Actores:** | Consultor |
| **Resumen:** | El caso de uso posibilita el acceso a la información relativa a los profesores y su estado en el sistema. |
| **Precondiciones:** | El usuario debe estar registrado y tener los permisos necesarios para realizar la acción. |
| **Referencias** | R16 |
| **Prioridad** | Auxiliar. |

## 2.11 Conclusiones

En este capítulo se realizó un análisis de los procesos que intervienen en la gestión de los profesores, lo que permitió obtener el conjunto de funcionalidades que debe cumplir el módulo, expresados en los requisitos funcionales y agrupados finalmente en los casos de uso del sistema. Esto constituye el inicio del desarrollo de la propuesta de solución y a partir de este punto se puede continuar la construcción de la misma.

# CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

## 3.1. Introducción

En el presente capítulo se efectuará el modelado del análisis y del diseño del sistema, los cuales son muy importantes en el desarrollo de software ya que constituye la vista lógica de la arquitectura. Por lo que se ajusta el resultado de estos modelos a las tecnologías y lenguajes que serán utilizados. Se muestran las realizaciones de los casos de uso críticos y secundarios, definidos en el capítulo anterior y que van a ser incluidos en la primera iteración del sistema, mediante los diagramas de clases del análisis y los diagramas de clases del diseño, expresados en diagramas de clases Web.

## 3.2 Análisis

"Durante el análisis, se analizan los requisitos que se describen en la captura de requerimientos, refinándolos y estructurándolos. El objetivo de hacerlo es conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que ayude a estructurar el sistema entero, incluyendo su arquitectura". (10)

El análisis abarca las abstracciones primarias de clases y objetos, así como los mecanismos presentes en el dominio del problema, o sea del mundo real, y no se consideran clases que definen detalles y soluciones en el sistema de software. Las clases que se modelan refinan los requisitos funcionales obtenidos anteriormente, son identificadas con sus relaciones y descritas en un diagrama de clases utilizando los siguientes estereotipos:

- Clase interfaz: Modelan la interacción entre el sistema y sus actores.
- Clase controladoras: Coordinan la realización de uno o unos pocos casos de uso, relacionando las actividades de los objetos que implementan sus funcionalidades.

- Clases entidad: Modelan información que posee larga vida y que es a menudo persistente.

### 3.2.1 Diagrama de clases del análisis.

En el diagrama de clases del análisis se representan los conceptos en un dominio del problema, representa las cosas del mundo real, no de la implementación automatizada.

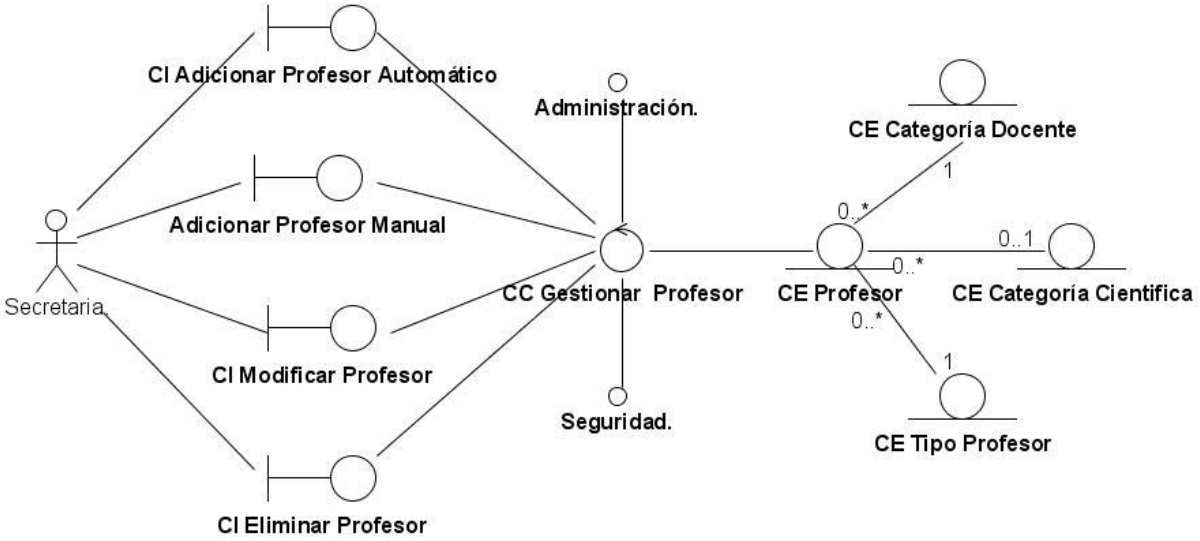


*Figura 3 Diagrama de clases del análisis CU Gestionar Profesor*

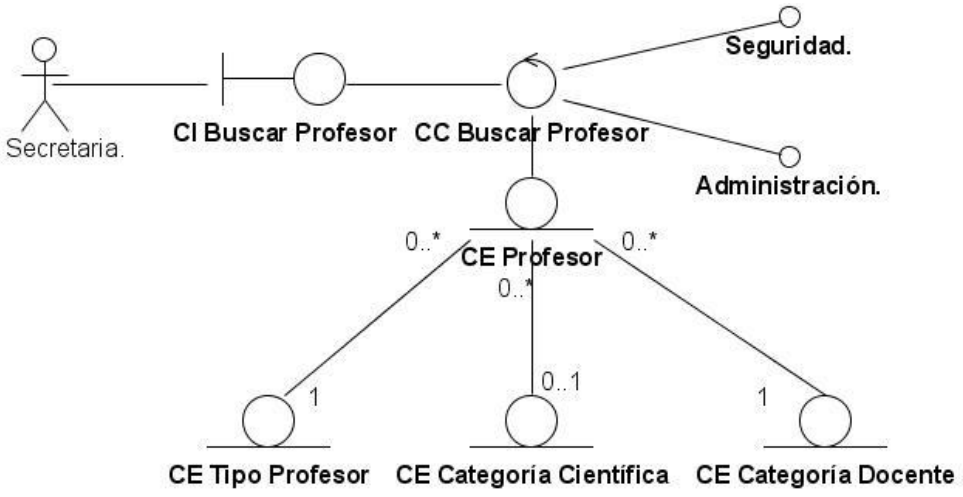


*Figura 4 Diagrama de clases del análisis CU Buscar Profesor.*

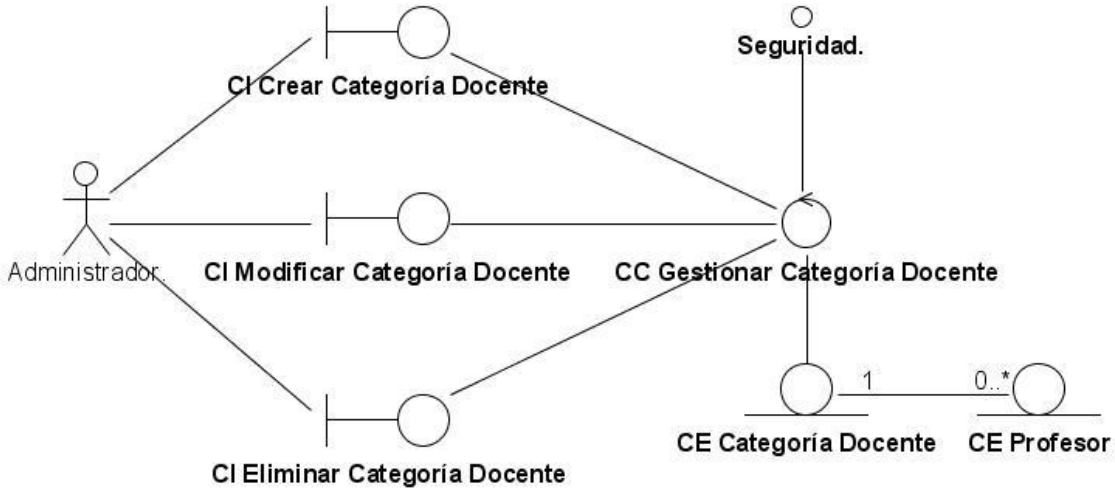


*Figura 5 Diagrama de clases del análisis CU Gestionar Categoría Docente*

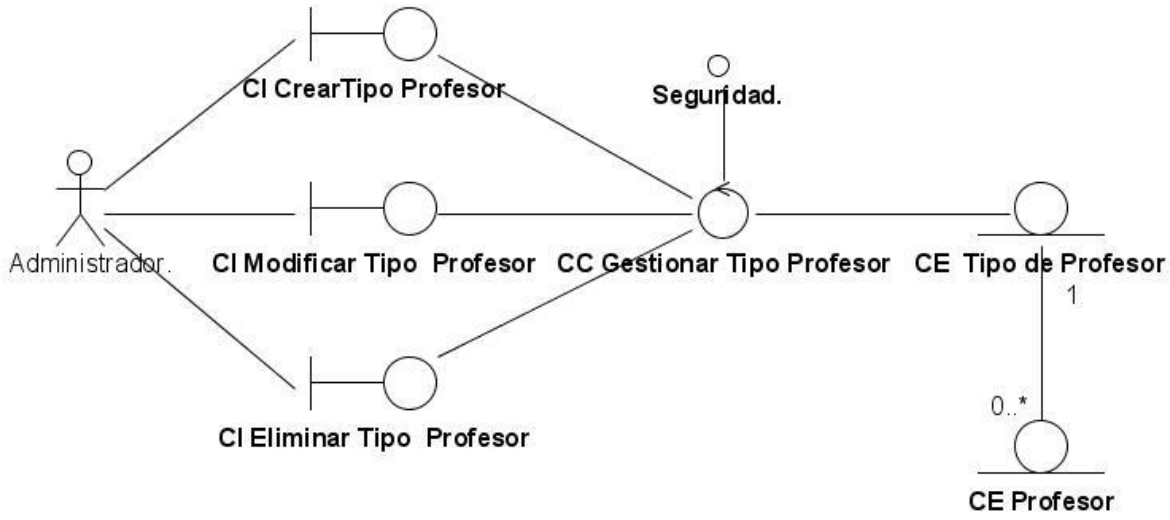


*Figura 6 Diagrama de clases del análisis CU Gestionar Tipo Profesor.*

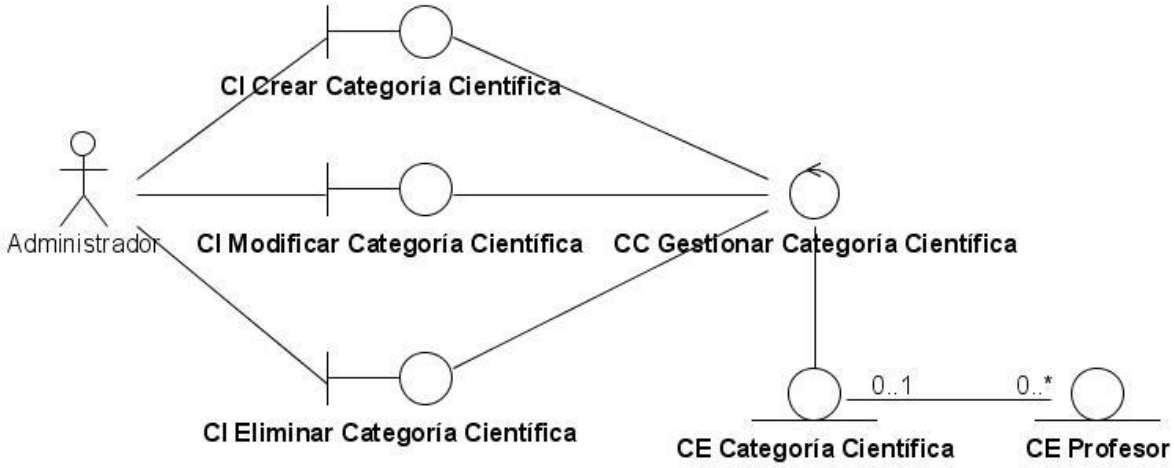


*Figura 7 Diagrama de clases del análisis CU Gestionar Categoría Científica.*

### 3.2.2 Diagramas de colaboración

Los diagramas de interacción representan una vista dinámica del sistema y se pueden clasificar en dos tipos, de Colaboración y de Secuencia. Un diagrama de interacción representa la secuencia de acciones que ocurren cuando el actor comienza el caso de uso, y los mensajes que se envían entre cada una de las clases. En el análisis se usan los diagramas de colaboración, ya que el objetivo principal es identificar las funcionalidades de cada objeto y las responsabilidades sobre ellos.
A continuación se muestra como ejemplo un diagrama de colaboración y en el Anexo 4 se pueden consultar los diagramas restantes para cada uno de los escenarios de los casos de usos.

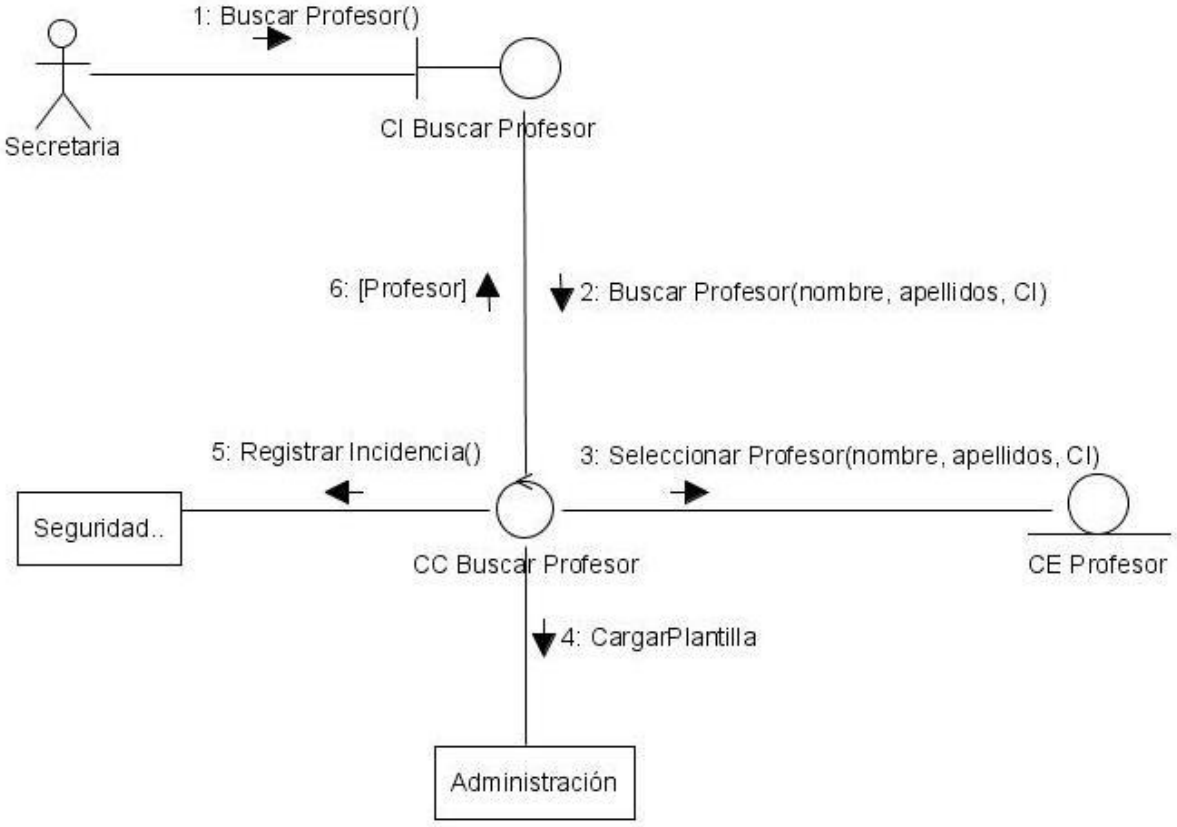


*Figura 8 Diagrama de colaboración Buscar Profesor*

### 3.3 Diseño

En el diseño el resultado del análisis es expandido a una solución técnica ya que se agregan nuevas clases que proveen de la infraestructura técnica: interfaces de usuario, manejo de bases de datos para almacenar objetos en una base de datos, comunicaciones con otros sistemas, etc. El diseño resulta en especificaciones detalladas para la fase de programación. Se modela el sistema de manera que soporte todos los requerimientos, incluyendo a diferencia del análisis, los requerimientos no funcionales.

#### 3.3.1 Patrones de diseño

Los patrones son soluciones a problemas recurrentes que ocurren una y otra vez en nuestro entorno. Se pueden considerar como recetas para solucionar varias veces un

problema del mismo tipo. Los desarrolladores lo usan como una forma de reutilizar la experiencia, clasificando las soluciones con términos de común denominación.

En el diseño de la propuesta de solución se tiene en cuenta un patrón clásico del diseño web conocido como arquitectura MVC (modelo-vista-controlador) que implementa Symfony. MVC está formado por tres niveles:

- El modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio. Se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación.
- La vista transforma el modelo en una página web que permite al usuario interactuar con ella.
- El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista. Mantiene aislado al modelo y a la vista de los detalles del protocolo utilizado para las peticiones.

De esta forma se separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones.

Una aplicación que utilice la arquitectura MVC separa el código del programa en tres capas, según su naturaleza. La lógica relacionada con los datos se incluye en el modelo, el código de la presentación en la vista y la lógica de la aplicación en el controlador. Se pueden utilizar otros patrones de diseño de forma tal que las capas puedan ser subdivididas en otras.

- Capa del modelo: Se puede dividir en la capa de acceso a los datos y en la capa de abstracción de la base de datos. De esta forma se logra crear una independencia entre las funciones para el acceso a los datos y el sistema gestor de base de datos, de manera tal que si el sistema cambia solo sea necesario modificar la capa de abstracción de la base de datos.

- Capa de la vista: Generalmente en una aplicación web las páginas suelen contener elementos que se muestran a lo largo de toda la aplicación de la misma forma y solo cambia el interior de las mismas. Esto permite que se pueda dividir la vista en un layout y en una plantilla, de forma tal que el layout sea global para toda la aplicación o a para un grupo de páginas específicas y la plantilla solo muestre las variables que define el controlador.
- Capa del controlador: En una aplicación web el controlador suele tener una gran carga de trabajo ya que debe encargarse del manejo de las peticiones del usuario, la seguridad, cargar la configuración de la aplicación y otras tareas similares. Por este motivo el controlador se puede dividir en un controlador frontal, que es único para cada aplicación, y las acciones, que incluyen el código específico del controlador de cada página. Utilizar un controlador frontal permitirá tener un único punto de entrada para toda la aplicación lo que hace más fáciles las configuraciones de acceso al sistema.

  De esta forma serían necesarios siete componentes para crear una página. Symfony permite crear aplicaciones rápidas y sencillas tomando lo mejor de MVC y simplificando el proceso de la siguiente forma:

  - El controlador frontal y el layout son comunes para todas las acciones de la aplicación.
  - El controlador frontal solo tiene código relativo al MVC, por lo que Symfony lo genera automáticamente y no es necesario crear uno.
  - Symfony contiene una librería llamada Propel que se encarga de generar automáticamente las clases de la capa del modelo, creando la estructura básica de las mismas y generando el código necesario.
  - La librería Propel permite también una rápida manipulación de datos ya que al encontrar restricciones de claves foráneas (o externas) o cuando encuentra datos de tipo fecha, crea métodos especiales para acceder y modificar los mismos.
  - Symfony cuenta con un componente llamado Creole que se encarga de hacer transparente para el programador la capa de abstracción de la

base de datos, de forma tal que si se modifica el sistema gestor de base de daros no será necesario modificar ninguna línea de código sino que solo se tendría que modificar un parámetro en un archivo de configuración.

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

En la realización del diseño se utilizaron además patrones GRASP, los cuales describen los principios fundamentales de la asignación de responsabilidades a objetos y patrones GOF.

**Patrones GRASP implementados**

- Experto: Este patrón se tiene en cuenta para la asignación de responsabilidades a las clases de forma tal que las mismas contengan la información necesaria para poder ejecutar una acción específica. Symfony contribuye a este patrón, ya que a través del uso de la librería externa Popel realiza la capa de abstracción en el modelo encapsulado toda la lógica de los datos y generando las clases con todas las funcionalidades comunes de las entidades. La lógica agregada a las mismas está en correspondencia con lo que planeta el patrón. El uso de este patrón permitirá a los objetos valerse de su propia información para hacer lo que se les pide, favorece la existencia de mínimas relaciones entre las clases, lo que permite contar con un sistema robusto y fácil de mantener.

- Creador: Este patrón se tiene en cuenta para la asignación de responsabilidades a las clases relacionadas con la creación de objetos, de forma tal que una instancia de un objeto solo pueda ser creada por el objeto que contiene la información necesaria para ello. El uso de este patrón permite crear las dependencias mínimas necesarias entre las clases, lo que favorece al mantenimiento del sistema y ofrece mejores oportunidades de reutilización.

- Alta cohesión: Este patrón se tiene en cuenta para realizar un diseño que evite contener clases con un alto grado de abstracción, que asuman responsabilidades que podían haber delegado a otros objetos o que tengan responsabilidades muy complejas. Se diseñaron las clases de forma tal que contengan las mínimas responsabilidades necesarias y colaboren con otras para llevar a cabo una tarea. Este patrón permitirá tener clases fáciles de mantener, de entender y reutilizar.

- Controlador: Este patrón se tiene en cuenta para realizar las asignaciones en cuanto al manejo de los eventos del sistema y definir sus operaciones. Symfony contribuye a la utilización de este patrón ya que define un controlador frontal que es el punto de acceso único a la aplicación y quien maneja las peticiones de los usuarios invocando a las operaciones necesarias para satisfacer las mismas.

- Bajo Acoplamiento: El acoplamiento mide la fuerza con que una clase está conectada a otra, de esta forma una clase con bajo acoplamiento debe tener un número mínimo de dependencia con otras clases. El uso de los patrones Experto y Creador contribuyen al bajo acoplamiento entre las clases del sistema. Este patrón se tuvo en cuenta por la importancia que significa realizar un diseño de clases independientes que puedan soportar los cambios de una manera fácil y permitan la reutilización.

**Patrones GOF implementados**

- Decorator: Este patrón permite añadir funcionalidad a una clase dinámicamente. Symfony implementa este patrón a través de la utilización de un loyout o plantilla global, que almacena el código HTML que es común a todas las páginas de la aplicación, para no tener que repetirlo en cada página. De esta forma el contenido de las plantillas específicas de un módulo se integra en el layout, o si se mira desde el otro punto de vista, el layout decora la plantilla.

- Singleton: Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Symfony define un objeto que guarda una referencia a todos los objetos del núcleo de Symfony relacionados con una petición dada, y ofrece un método de acceso único para cada uno de ellos.

- Facade: Este patrón permite utilizar una interfaz común para un conjunto de interfaces del módulo, haciendo que éste sea más fácil de usar.

### 3.3.2 Diagrama de clases del diseño

La forma tradicional de modelar las clases del diseño, no es factible a la hora de diseñar una aplicación Web. Por ese motivo, se utiliza una extensión de UML para Web, que se adapta a la arquitectura de este tipo de sistemas.
Para obtener un nivel correcto de abstracción y detalle se modelan los artefactos del sistema como las páginas, los enlaces entre las mismas, todo el código que irán creando, así como el contenido dinámico de estas una vez que estén en el navegador del cliente; estos son los artefactos que se necesitan modelar para que el desarrollador los implemente luego y obtener así el producto final.

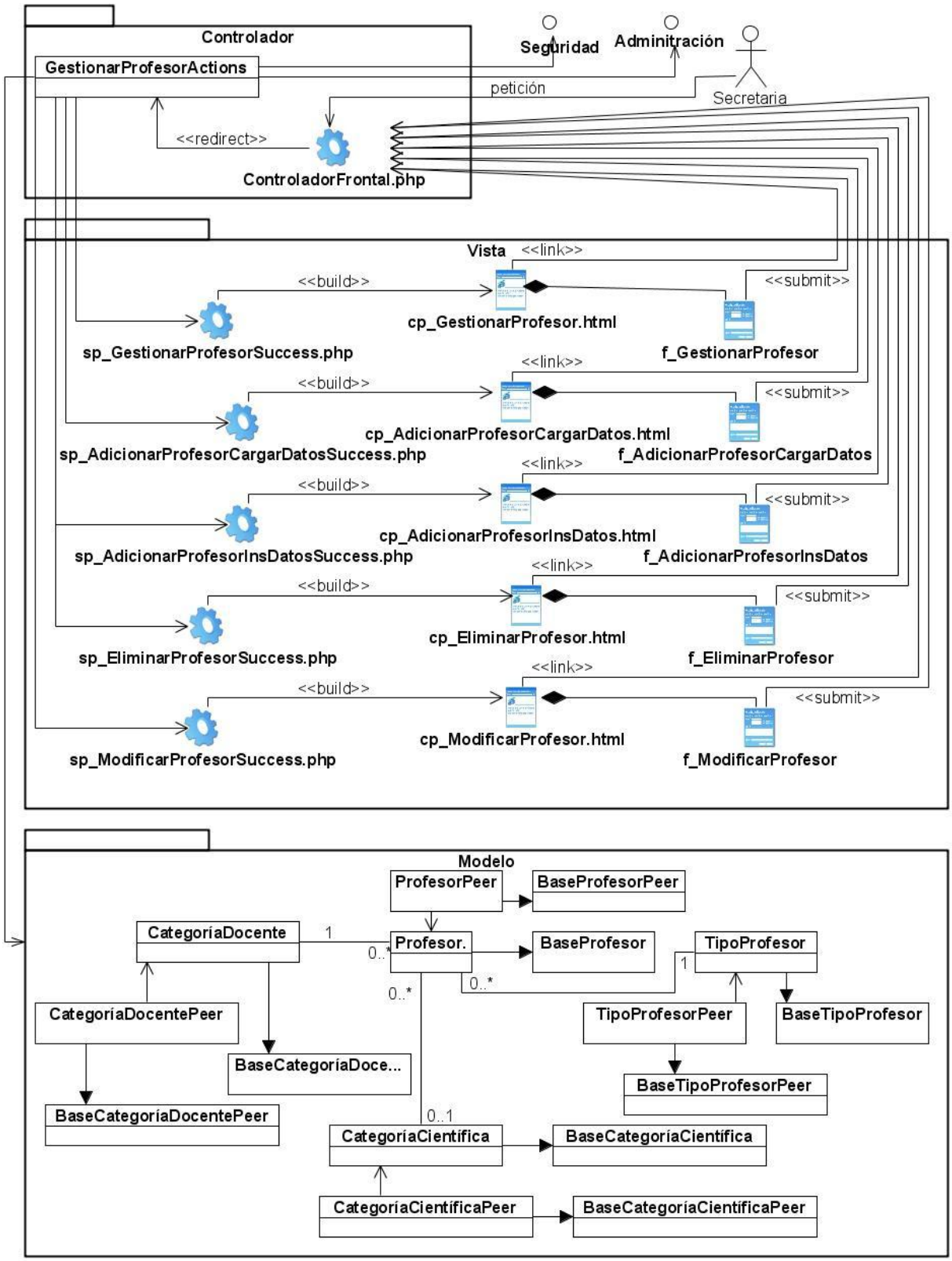


*Figura 9 Diagrama de clases del diseño CU Gestionar Profesor*

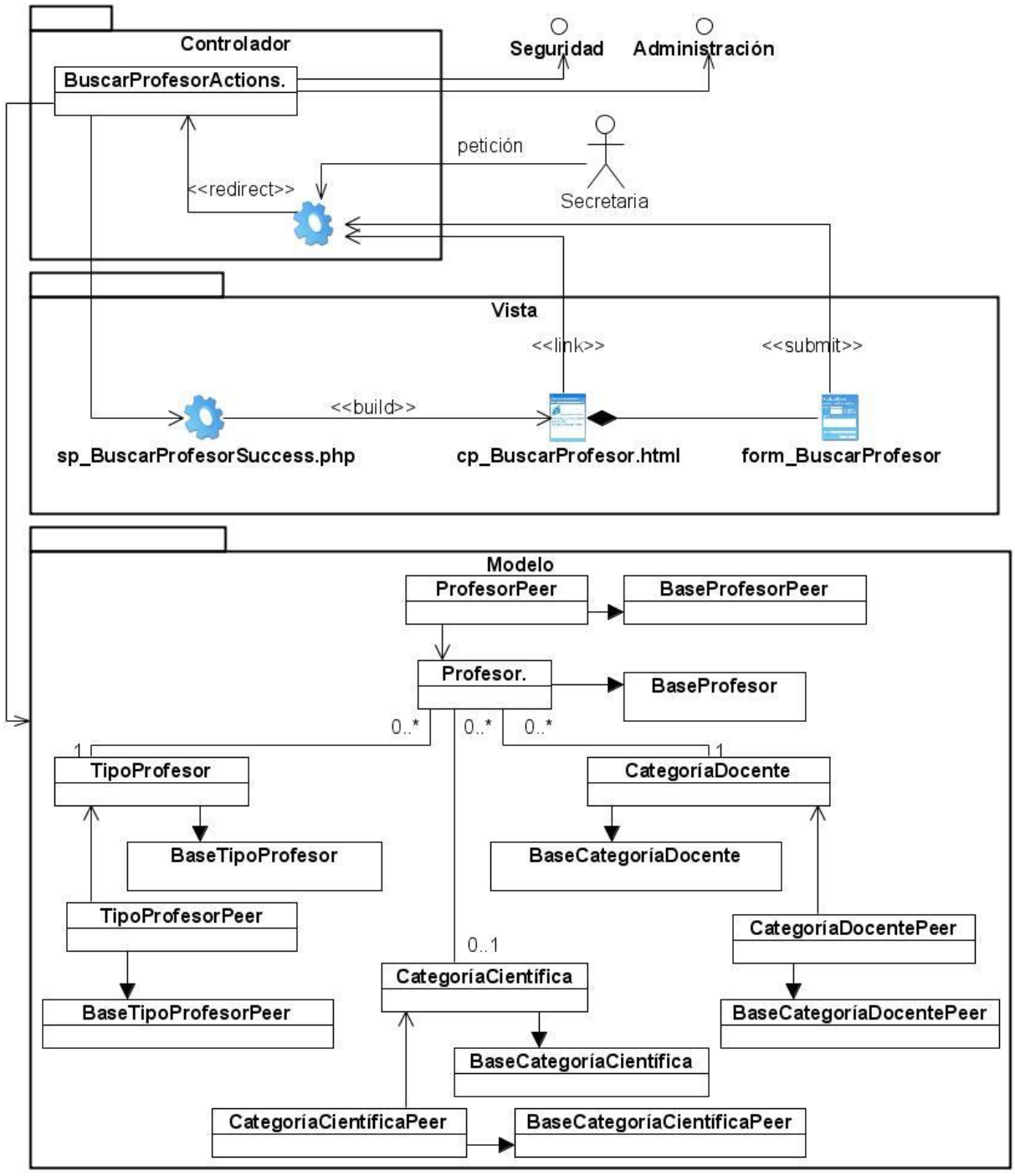


*Figura 10 Diagrama de clases del diseño CU Buscar Profesor*

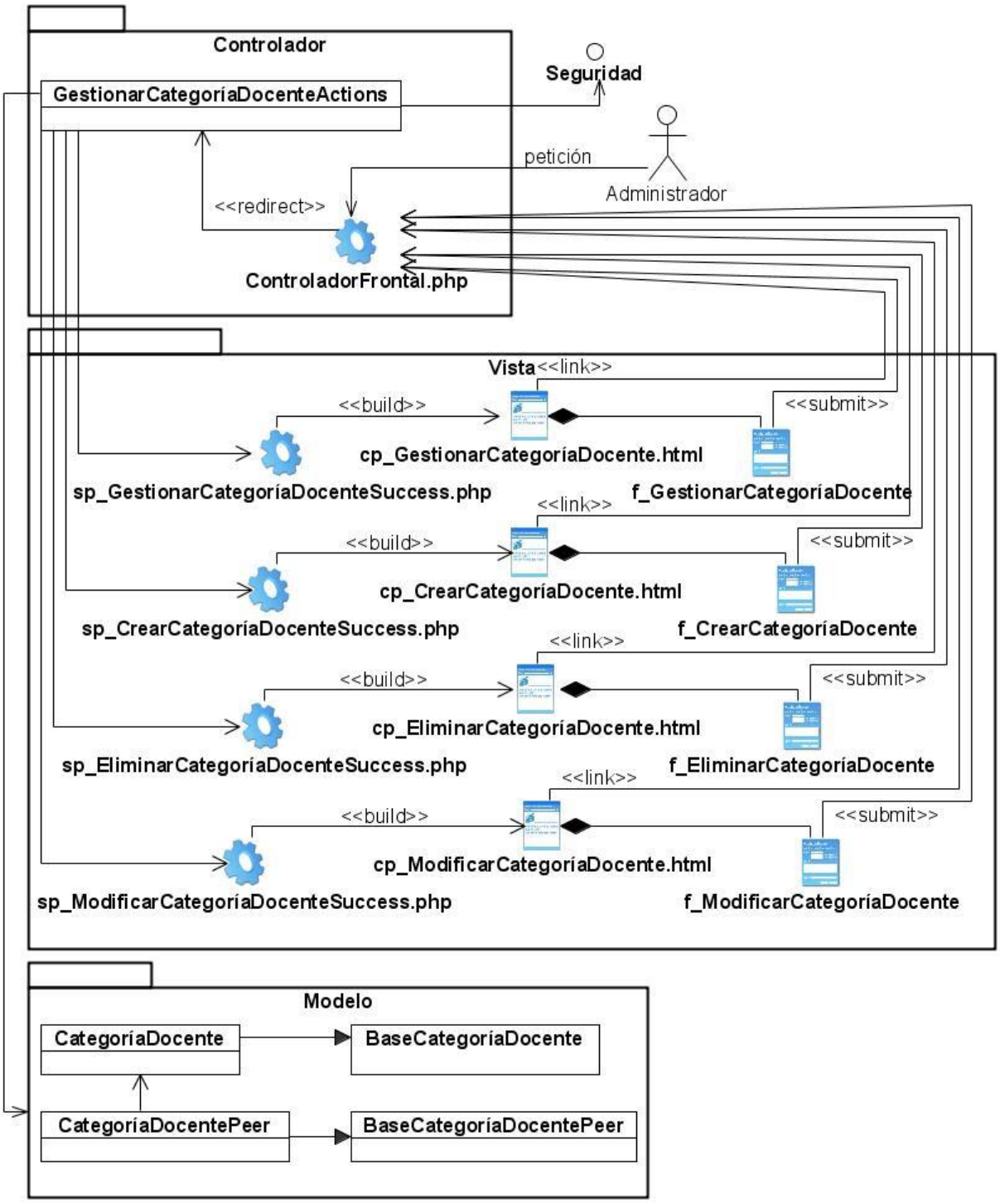


*Figura 11 Diagrama de clases del diseño CU Gestionar Categoría Docente*

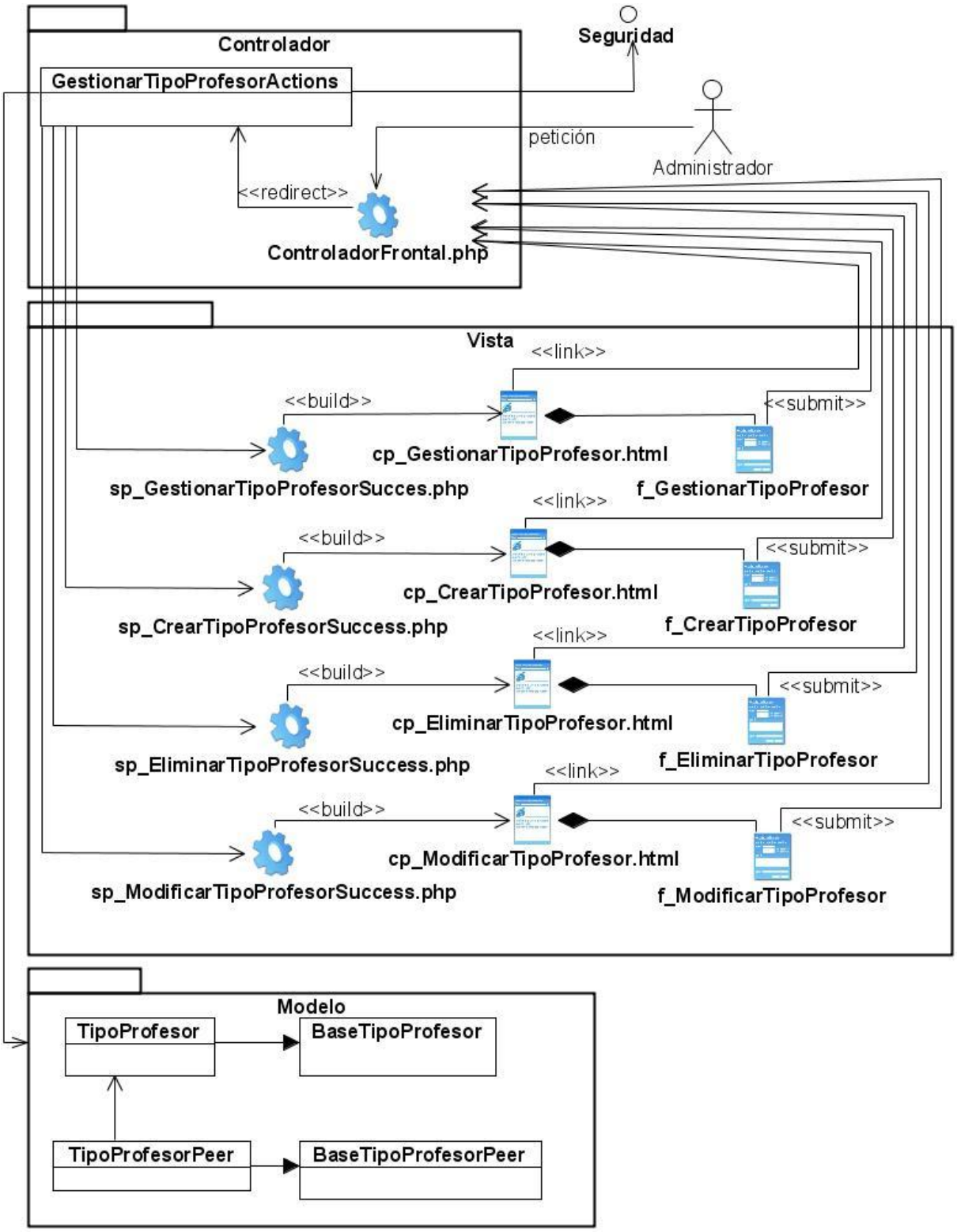


*Figura 12 Diagrama de clases del diseño CU Gestionar Tipo Profesor*

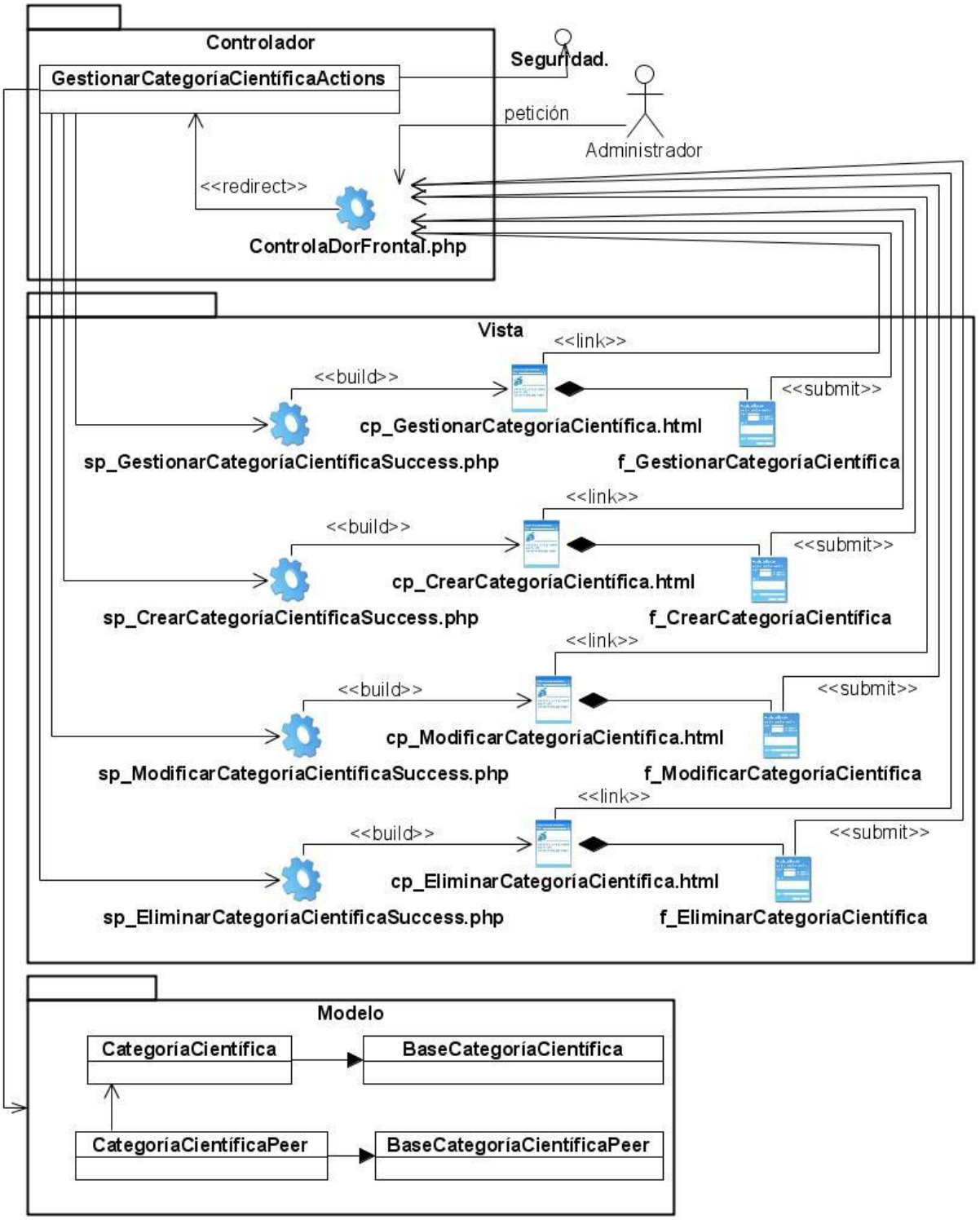


*Figura 13 Diagrama de clases del diseño CU Gestionar Categoría Científica*

### 3.3.3 Diagramas de secuencia

Los diagramas de secuencia forman parte del modelado dinámico del sistema y proporcionan una vista detallada de los casos de uso. Son diagramas que muestran la interacción organizada de objetos, mediante mensajes que se envían entre sí, en una secuencia de tiempo. Son útiles para observar la vida de los objetos en un sistema, identificar llamadas a realizar o posibles errores del modelado estático que imposibiliten el flujo de información.

A continuación se muestra como ejemplo un diagrama de secuencia y en el Anexo 5 se pueden consultar los diagramas restantes para cada uno de los escenarios de los casos de usos.

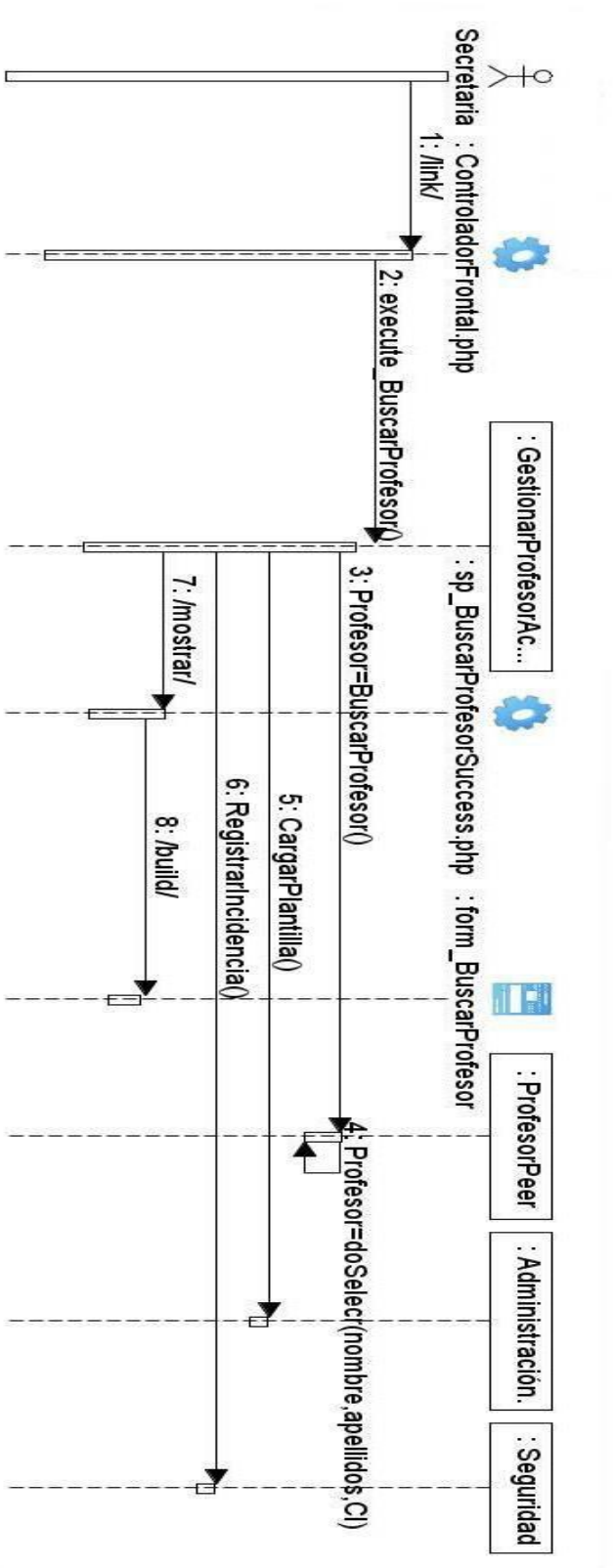


*Figura 14 Diagrama de secuencia Buscar Profesor*

### 3.4 Diseño de la base de datos

Para el diseño de la base de datos se utilizaron los modelos físicos y lógicos de datos, a través de los cuales se diseñaron las clases persistentes con sus relaciones y se modeló la distribución de las tablas en la base de datos. El diseño propuesto satisface las necesidades de persistencia de los datos que el módulo requiere en cumplimiento de sus requerimientos funcionales y permite la integración con el resto del sistema.

#### 3.4.1 Modelo lógico

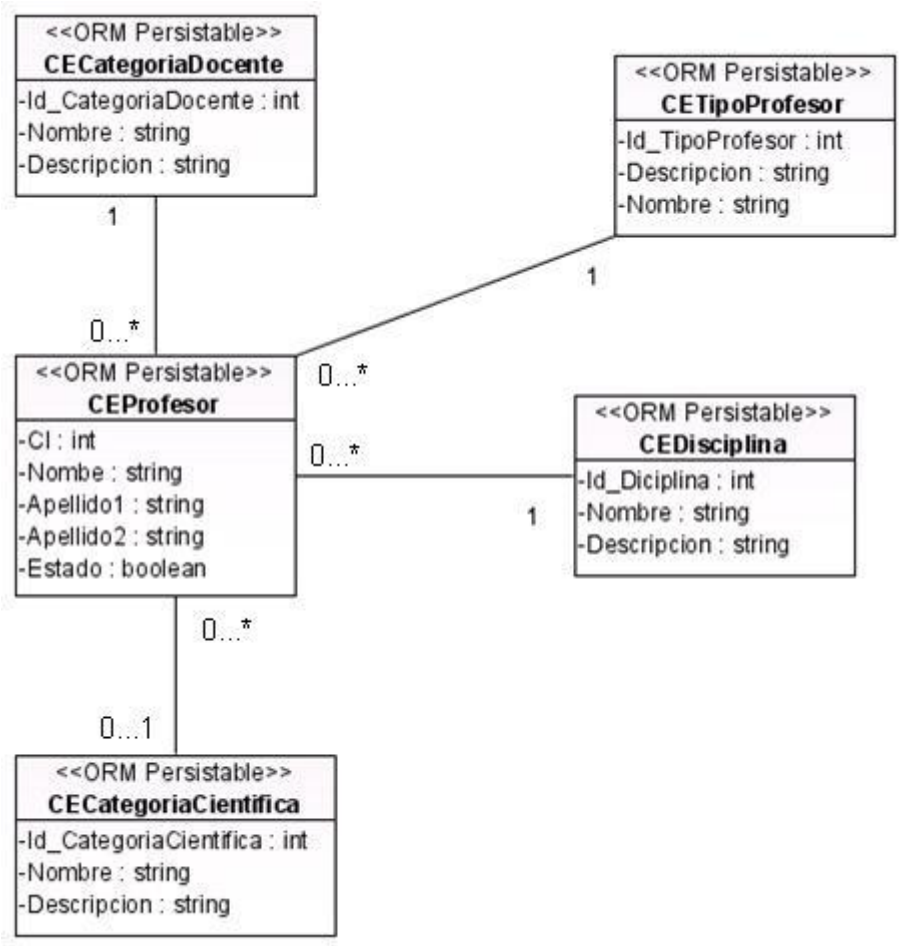


*Figura 15 Modelo lógico de datos.*

### 3.4.2 Modelo físico

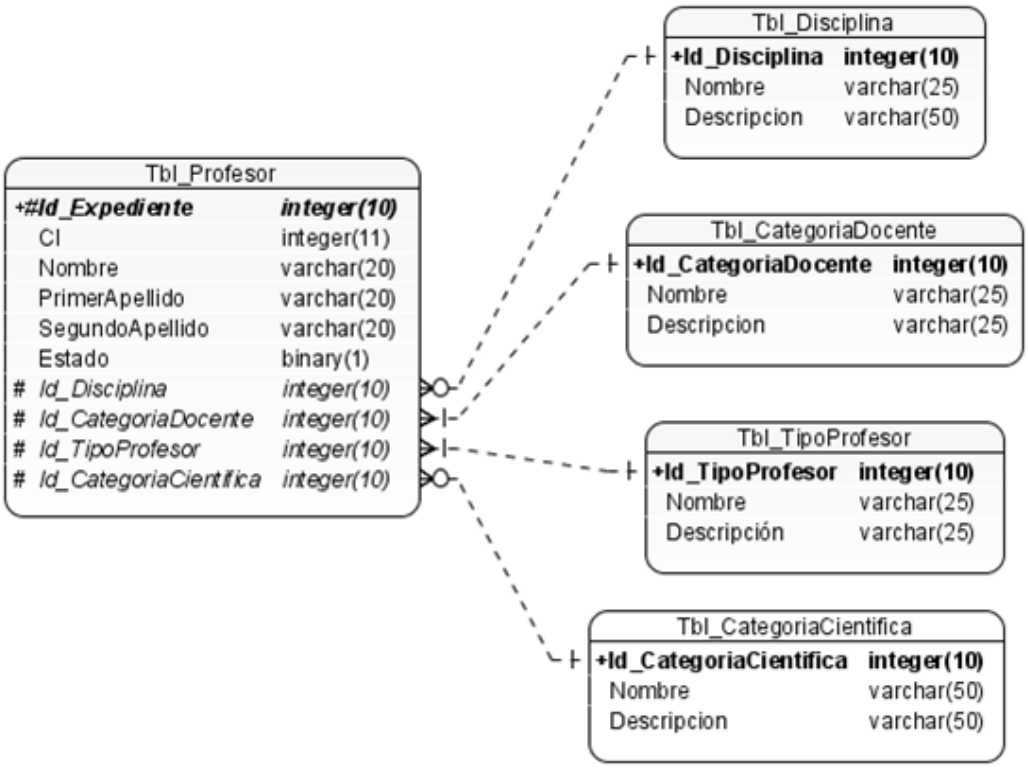


*Figura 16 Modelo físico de datos*

### 3.5 Diagrama de despliegue

A continuación se muestra la distribución física del sistema teniendo en cuenta la arquitectura del software y del hardware.

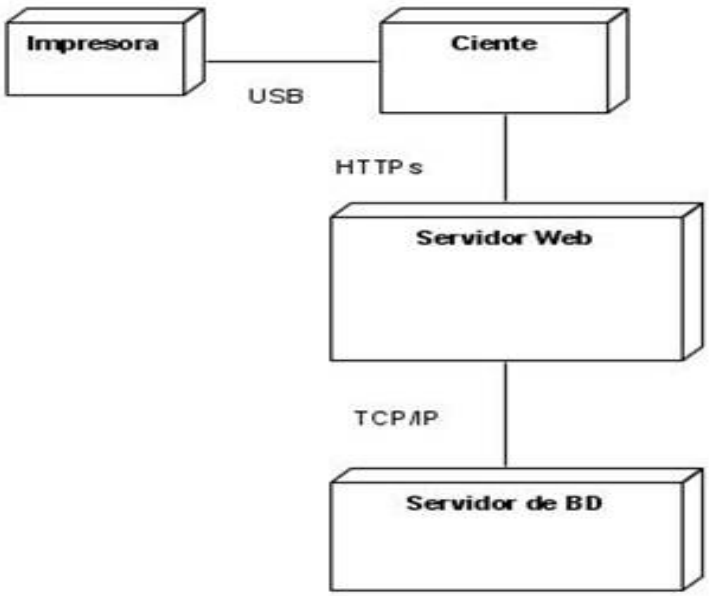


*Figura 17 Diagrama de despliegue.*

### 3.6 Conclusiones

En este capítulo se finaliza la etapa de análisis y diseño del sistema obteniendo un modelo más detallado de la solución propuesta. Se realizaron los diagramas de clases del análisis y de clases del diseño de los casos de uso críticos del sistema así como los diagramas de colaboración y secuencia para cada una de las funcionalidades descritas, lo que permite una idea más específica del sistema que se propone. Se describe la arquitectura y los patrones que se tuvieron en cuenta para el diseño de los casos de uso. Se muestran además los modelos físicos y lógicos de datos que satisfacen las necesidades del módulo y que son de gran importancia para la implementación del mismo, así como el diagrama de despliegue.

# CAPÍTULO 4: ANÁLISIS DE COSTO

## 4.1 Introducción

La planificación de proyectos tiene como principal objetivo establecer planes razonables para realizar las actividades de ingeniería del software y manejar los cambios de los proyectos de software. Esta actividad incluye la estimación de los resultados del proyecto y los valores de costo, tiempo y recursos requeridos. En el presente capítulo se realiza una estimación en término del capital humano que permitirá considerar el tiempo necesario para la entrega del módulo, fechas de terminación para cada etapa de desarrollo, etc. Para lograr este propósito se utilizará el método de estimación por puntos de casos de uso el cual permite predecir el tamaño de un sistema a partir de las características de sus requisitos expresados en los casos de uso.

## 4.2 Estimación del esfuerzo

Método a utilizar: Estimación por Puntos por Caso de Uso.
El primer paso para la estimación consiste en el cálculo de los Puntos de Casos de Uso sin ajustar.

Se calcula a partir de la siguiente ecuación:
**UUCP = UAW + UUCW donde,**
UUCP: Puntos de Casos de Uso sin ajustar
UAW: Factor de Peso de los Actores sin ajustar
UUCW: Factor de Peso de los Casos de Uso sin Ajustar

**Factor de peso de los actores sin ajustar (UAW)**
Este factor se calcula mediante un análisis de la cantidad de actores presentes en el sistema y la complejidad de cada uno de ellos, para lo cual se tiene en cuenta, en primer lugar si el actor es una persona o un sistema, y en segundo lugar, la forma en la que el actor interactúa con el sistema.

*Tabla 3.1 Factor de peso de los actores sin ajustar*

| Actores | Descripción | Complejidad | Factor de Peso |
|---|---|---|---|
| Secretaria Docente. | Una persona que interactúa con el sistema mediante una interfaz gráfica. | Complejo | 3 |
| Administrador | | | |

UAW = Sumatoria de la multiplicación de la cantidad de actores de un tipo con su factor de peso.
Cantidad de actores de tipo complejo: 3
UAW = 2x3
UAW = 6

**Factor de peso de los casos de uso sin ajustar (UUCW)**

Este valor se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos, para lo cual se tiene en cuenta la cantidad de transacciones que presenta el caso de uso. Una transacción es una secuencia de actividades atómica, está representada por uno o más pasos del flujo de eventos principal del Caso de Uso, pudiendo existir más de una transacción dentro del mismo Caso de Uso.

*Tabla 3.2 Factor de peso de los casos de uso sin ajustar*

| Caso de Uso | Descripción | Complejidad | Factor de Peso |
|---|---|---|---|
| CUS Gestionar Profesor. | Contiene 3 transacciones. | Simple | 5 |
| CUS Buscar Profesor. | Contiene 2 transacciones. | | |
| CUS Gestionar Tipo Profesor. | Contiene 3 transacciones. | | |
| CUS Gestionar Categoría Docente. | Contiene 3 transacciones. | | |

| | | | |
|---|---|---|---|
| CUS Gestionar Categoría Científica. | Contiene 3 transacciones. | | |

UUCW = Casos de uso del sistema y su complejidad.

UUCW = 5 x 5

UUCW = 25

Finalmente, los Puntos de Casos de Uso sin ajustar resultan

UUCP = UAW + UUCW

UUCP = 6 + 25

UUCP = 31

El segundo paso es calcular los puntos de Casos de Uso ajustados.

**UCP = UUCP x TCF x EF**

Donde,

UCP: Puntos de Casos de Uso ajustados

UUCP: Puntos de Casos de Uso sin ajustar

TCF: Factor de complejidad técnica

EF: Factor de ambiente

**Factor de complejidad técnica (TCF)**

Para calcular este coeficiente se analizan una serie de factores que pueden determinar la complejidad técnica del sistema, a estos factores se les asocia un valor de 0 a 5 que determina el vínculo del mismo con las características deseadas para del sistema

*Tabla 3.3 Factor de complejidad técnica.*

| Factor | Descripción | Peso | Valor asignado | Peso*Valor |
|---|---|---|---|---|
| T1 | Sistema distribuido. | 2 | 0 | 0 |
| T2 | Objetivos de performance o tiempo de respuesta. | 1 | 4 | 4 |
| T3 | Eficiencia del Usuario Final. | 1 | 2 | 2 |
| T4 | Procesamiento interno complejo. | 1 | 3 | 3 |
| T5 | El código debe ser reutilizable. | 1 | 5 | 5 |

| | | | | |
|---|---|---|---|---|
| T6 | Facilidad de instalación. | 0.5 | 4 | 2 |
| T7 | Facilidad de uso. | 0.5 | 3 | 1.5 |
| T8 | Portabilidad. | 2 | 4 | 8 |
| T9 | Facilidad de cambio. | 1 | 3 | 3 |
| T10 | Concurrencias. | 1 | 5 | 5 |
| T11 | Incluye objetivos específicos de seguridad. | 1 | 5 | 5 |
| T12 | Provee accedo directo a terceras partes. | 1 | 0 | 0 |
| T13 | Se requieren facilidades especiales de entrenamiento a usuarios. | 1 | 3 | 3 |

El Factor de complejidad técnica resulta:

TCF = 0.6 + 0.01 x Σ (Pesoi x Valor asignadoi)

TCF=0.6+0.01x (0+4+2+3+5+2+1.5+8+3+5+5+0+3)

TCF=0.6+0.01x 41.5

TCF=1.015

**Factor de ambiente (EF)**

Para calcular este coeficiente se analizan una serie de factores que pueden determinar el tiempo requerido para el desarrollo del sistema, teniendo en cuenta aspectos como habilidades, conocimientos, etc. de los involucrados en la realización del sistema. A estos factores se les asocia un valor de 0 a 5 que determina el vínculo del mismo con las características deseadas para del sistema.

*Tabla 3.4 Factor ambiente.*

| Factor | Descripción | Peso | Valor asignado | Peso* Valor |
|---|---|---|---|---|
| E1 | Familiaridad con el modelo de proyecto utilizado. | 1.5 | 0 | 0 |
| E2 | Experiencia en la aplicación. | 0.5 | 3 | 1.5 |
| E3 | Experiencia en orientación a objetos. | 1 | 4 | 4 |

| | | | | |
|---|---|---|---|---|
| E4 | Capacidad del analista líder | 0.5 | 5 | 2.5 |
| E5 | Motivación. | 1 | 4 | 4 |
| E6 | Estabilidad de los requerimientos. | 2 | 2 | 4 |
| E7 | Personal part-time. | -1 | 0 | 0 |
| E8 | Dificultad del lenguaje de programación. | -1 | 2 | -2 |

El Factor de ambiente resulta:

EF =1.4 - 0.03 x Σ (Peso x Valor asignado)

EF =1.4 – 0.03 x Σ (0+1.5+4+2.5+4+4+0-2)

EF = 1.4 – 0.03 x 14

EF = 0.98

Finalmente, los Puntos de Casos de Uso ajustados resultan:

UCP = UUCP x TCF x EF

UCP = 31 x 1.015 x 0.98

UCP = 30,8357

El esfuerzo en horas-hombre viene dado por:

**E = UCP x CF**

Donde:

E= Esfuerzo

UCP= Puntos de Casos de Uso ajustados.

CF= Factor de conversión (para este tipo de proyecto 20 horas-hombre/Punto de Casos de Uso) rebaja

E = UCP x CF

E = 30,8357x 20

E = 616,714

Se considera que este esfuerzo representa un porcentaje del esfuerzo total del proyecto. Para una estimación más completa de la duración total del proyecto hay que agregar a la estimación del esfuerzo obtenida por los Puntos de Casos de Uso, las estimaciones de esfuerzo de las demás actividades relacionadas con el desarrollo de software.

Teniendo en cuenta los siguientes valores porcentuales para la distribución del esfuerzo entre las diferentes actividades de un proyecto, que estadísticamente se considera aceptable, se obtiene:

*Tabla 3.5 Distribución del esfuerzo*

| **Actividad** | **Porcentaje** | **Horas-Hombre** |
|---|---|---|
| Análisis | 10% | 154,1785 |
| Diseño | 20% | 308,357 |
| Programación. | 40% | 616,714 |
| Prueba. | 15% | 231,26775 |
| Sobrecarga (otras actividades.) | 15% | 231,26775 |
| Total | 100% | 1541,785 |

Esfuerzo Total (horas-hombres) 1442.3115.

Para la etapa de análisis y diseño se requiere de un esfuerzo de 462,5355 horas-hombres, si se considera que trabajan dos personas, 48 horas como promedio en la semana, esta etapa debe terminarse en aproximadamente 5 semanas.

## 4.3 Conclusiones

La realización de la estimación de costos ha servido para establecer cronogramas y estimaciones razonables y ha permitido una mejor organización y control de las tareas establecidas durante el desarrollo del módulo. Esta estimación solo tendría un valor económico si se pudiera multiplicar por el costo de un estudiante en la UCI, por lo que la misma solo tiene un valor referencial en cuanto al factor humano de tiempo-trabajo. Teniendo en cuenta la estimación realizada y los beneficios tangibles que se podrán obtener al concluir el desarrollo del módulo, entre los que se destacan el uso más eficiente del tiempo de trabajo de las secretarias y la accesibilidad a la información almacenada de los profesores, y considerando un grupo de factores que ofrece la Universidad cuya utilización no genera costo alguno, como la infraestructura tecnológica instalada, la correspondencia entre el proyecto a desarrollar y el propósito del centro así como la necesidad de poner en práctica las capacidades de los estudiantes, podemos concluir que la realización del módulo Profesor resulta factible.

# CONCLUSIONES

Para el desarrollo de esta investigación se realizó un estudio bibliográfico que permitió tener un conocimiento más profundo de la situación actual y las tendencias de los sistemas de gestión académica. A través del mismo, se demostró la necesidad de desarrollar un módulo capaz de automatizar los procesos de gestión de los profesores para la nueva versión del sistema de gestión académica Akademos. La automatización de estos procesos debe permitir que el sistema pueda ser aplicado en diferentes centros educacionales.

La propuesta que se hace en este trabajo para la nueva versión del Módulo Profesor del Sistema de Gestión Académica Akademos, permite gestionar la información referente a los profesores de un centro de estudios y acceder a la misma a través de un grupo de reportes. Ofrece la posibilidad de buscar a un profesor dentro del sistema, lo que permitirá realizar este trabajo de una forma más sencilla y rápida. Uno de los aspectos más novedosos de esta propuesta, es que se tuvo en cuenta permitir la configuración de algunos elementos que pueden ser diferentes en dependencia del centro donde se utilice el sistema, de forma tal que se facilite su uso. Las funcionalidades que se tuvieron en cuenta para el desarrollo de esta propuesta permitieron obtener el diseño de un módulo encaminado a eliminar las dificultades presentadas por el actual módulo del sistema.

El desarrollo del trabajo con la metodología RUP permitió documentar el mismo desde sus inicios, lo que servirá como base de estudio para futuros miembros del equipo de trabajo permitiendo de esta forma una comprensión más rápida y fácil de todas las etapas de concepción del módulo. La propuesta de análisis y diseño obtenida, constituye una base para el trabajo de los implementadores del módulo así como para futuras iteraciones o ciclos de desarrollo.

# RECOMENDACIONES

A partir del estudio realizado en este trabajo y teniendo en cuenta un grupo de ideas que surgieron durante el desarrollo del análisis y diseño del módulo, se considera que se puede recomendar a la dirección del proyecto Akademos las siguientes acciones a tener en cuenta de forma tal que se logre una mayor eficiencia:

- Continuar la investigación comenzada en este trabajo manteniendo un seguimiento sobre los sistemas de gestión académica que son utilizados en el mundo, para garantizar mejoras en las próximas fases de desarrollo del módulo.

- Realizar el análisis y diseño de los casos de uso del sistema que no se tuvieron en cuenta para el desarrollo de esta iteración, considerando la importancia que tienen para la utilización del módulo en los centros educacionales.

- Implementar los casos de uso propuestos para esta iteración utilizando el análisis y diseño obtenido en este trabajo.

- Considerar la posibilidad de incluir en el sistema funcionalidades que permitan gestionar el currículo de un profesor así como los eventos en los que ha participado y las publicaciones realizadas por el mismo, teniendo en cuenta la importancia que representa conocer esta información de los docentes en un centro de estudio.

# BIBLIOGRAFÍA REFERENCIADA

1. CORDOBA, U. D. Sistema Integral de Gestión Académica. 2008, Disponible en: http://www.gestion.uco.es/gestion/aplicaciones/siga.

2. SOFT, K. Soluciones Informáticas para el Sector Docente: 25 razones para elegir AGORA. 2008, Disponible en: http://www.kherian.com/agora_porque.asp.

3. XHARDEZ, V. ¿Qué es el Proyecto Alba? 2006, Disponible en: http://www.proyectoalba.com.ar/spip.php?article3.

4. NOEL MIÑO, E. P. Análisis y diseño del Módulo Estudiante para el sistema de gestión Académica-akademos. 2007, Disponible en: http://bibliodoc.uci.cu/TD/TD_0325_07.pdf.

5. GONZÁLEZ, D. J. A. T.; DELGADO, L. Y. C., *et al.* INFORMÁTICA 2002 INFORMÁTICA EN LA EDUCACIÓN GESTACAD. SISTEMA PARA LA GESTIÓN ACADÉMICA.

6. MARRERO, D. P. C. J. L. A. O. Y. S. R. H. E. Manual de Usuario del SIGENU Sistema de Gestión de la Nueva Universidad. Abril del 2007,

7. MOLPECERES, A. Procesos de desarrollo: RUP, XP y FDD. 2003, Disponible en: http://www.javahispano.org/contenidos/es/procesos_de_desarrollo/.

8. POTENCIER, F. Z. Y. F. Symfony, la guía definitiva. 2008, Disponible en: http://www.librosweb.es/symfony/.

9. PRESSMAN, R. S. *Ingeniería del Software, un enfoque práctico.* Quinta ed. Editorial Félix Varela, 2002.

10. IVAR JACOBSON, G. B., JAMES RUMBAUGH. *El Proceso Unificado de Desarrollo del software.* Editorial Félix Varela, 2004. vol. 1,

# BIBLIOGRAFÍA CONSULTADA

1. CORDOBA, U. D. Sistema Integral de Gestión Académica. 2008, Disponible en: http://www.gestion.uco.es/gestion/aplicaciones/siga.

2. DAMIÁN CERVANTES RODÓN, R. A. F. Análisis y diseño del sistema de venta y admisión de la plataforma de servicios postales. . 2007, Disponible en: http://biblioteca.uci.cu/sbd/biuci/index.html.

3. FRANK BENAVIDES DALMENDRAY, D. C. R., NOEL MIÑO HERRERA, ELISABEL; PEREZ URBAY, A. S. H., DIANLY SANTILER ÁLVAREZ, NORGES SÁNCHEZ, et al. XVI Forum de Ciencia y Técnica. Akademos, Sistema Automatizado para la Gestión Académica. 2006.

4. GALVE, J. P. G. G. I. J. FUNDAMENTOS DE LA METODOLOGIA RUP RATIONAL UNIFIED PROCESS 2007, Disponible en: http://www.scribd.com/doc/297224/RUP.

5. GONZÁLEZ, D. J. A. T.; DELGADO, L. Y. C., et al. INFORMÁTICA 2002 INFORMÁTICA EN LA EDUCACIÓN GESTACAD. SISTEMA PARA LA GESTIÓN ACADÉMICA.

6 SINNOVA, G. S. Herramientas y Soluciones IBM. Rational Rose Enterprise. 2007, Disponible en: http://www.rational.com.ar/herramientas/roseenterprise.html.

7. IBM. Rational Rose Enterprise. 2008, Disponible en: http://www-142.ibm.com/software/dre/ecatalog/Detail.wss?locale=es_ES&synkey=M221280M46834Z27.

8. INFORMÁTICA, D. SIGA Sistema Integrado de Gestión Académica. 2008, Disponible en: http://www.dara.es/siga/.

9. INTERNATIONAL, V. P. Visual Paradigm for UML. 2008, Disponible en:

http://www.visual-paradigm.com/product/vpuml/.

10. IVAR JACOBSON, G. B., JAMES RUMBAUGH. El Proceso Unificado de Desarrollo del software. Editorial Félix Varela, 2004. vol. 1,

11. ---. El Proceso Unificado de Desarrollo del software. Editorial Félix Varela, 2004. vol. 2,

12. LARMAN, C. UML y Patrones. Introducción al análisis y diseño orientado a objetos. Editorial Félix Varela, 2004.

13. LATINA, O. Porque es importante UML. 2008, Disponible en: http://www.osmosislatina.com/lenguajes/uml/basico.htm.

14. MALDONADO, D. M. El proyecto Alba administrando la educación. 2008, Disponible en: http://www.aplicacionesempresariales.com/gestion/el-proyecto-alba-administrando-la-educacion.html.

15. MARRERO, D. P. C. J. L. A. O. Y. S. R. H. E. Manual de Usuario del SIGENU Sistema de Gestión de la Nueva Universidad. Abril del 2007,

16. MOLPECERES, A. Procesos de desarrollo: RUP, XP y FDD. 2003, Disponible en: http://www.javahispano.org/contenidos/es/procesos_de_desarrollo/.

17. NOEL MIÑO, E. P. Análisis y diseño del Módulo Estudiante para el sistema de gestión Académica-Akademos. 2007, Disponible en: http://bibliodoc.uci.cu/TD/TD_0325_07.pdf.

18. OLIVIA RODRIGUEZ ABRIL, D. S. A. Análisis y Diseño del módulo Reportes del sistema automatizado para la gestión académica. Akademos. 2007, Disponible en: http://biblioteca.uci.cu/sbd/biuci/index.html.

19. PHP, G. D. D. D. PHP Manual. 2008, Disponible en: http://www.php.net/manual/es/index.php.

20. POSTGRESQL. Advantages. Disponible en: http://www.postgresql.org/about/advantages.

21. POSTGRESQL, E. E. D. D. D. Tutorial de PostgreSQL. Disponible en: http://es.tldp.org/Postgresql-es/web/navegable/tutorial/tutorial.html.

22. POTENCIER, F. Z. Y. F. Symfony, la guía definitiva. 2008, Disponible en: http://www.librosweb.es/symfony/.

23. PRESSMAN, R. S. Ingeniería del Software, un enfoque práctico. Quinta ed. Editorial Félix Varela, 2002.

24. SANCHEZ, M. A. M. Metodologías De Desarrollo De Software. 2004, Disponible en: http://www.informatizate.net/articulos/pdfs/metodologias_de_desarrollo_de_software_07062004.pdf.

25. SOFT, K. Soluciones Informáticas para el Sector Docente: 25 razones para elegir AGORA. 2008, Disponible en: http://www.kherian.com/agora_porque.asp.

26. ---. Soluciones Informáticas para el Sector Docente: AGORA. 2008, Disponible en: http://www.kherian.com/mexico/default.asp.

27. WIKIPEDIA. Proceso Unificado de Rational. 2008, Disponible en: http://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational

28. XHARDEZ, V. ¿Qué es el Proyecto Alba? 2006, Disponible en: http://www.proyectoalba.com.ar/spip.php?article3.

# GLOSARIO DE TÉRMINOS

**Herramienta CASE:** Acrónimo de Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador, 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. Estas herramientas nos pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, calculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras.

**HTML:** Acrónimo inglés de HyperText Markup Language (lenguaje de marcas hipertextuales), lenguaje de marcación diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas web. Gracias a Internet y a los navegadores del tipo Internet Explorer, Opera, Firefox o Netscape, el HTML se ha convertido en uno de los formatos más populares que existen para la construcción de documentos y también de los más fáciles de aprender.

**HTTP:** Acrónimo del inglés HyperText Transfer Protocol, es el protocolo de transferencia de hipertexto usado en cada transacción de la Web

**HTTPS:** Es la versión segura del protocolo HTTP. El sistema HTTPS utiliza un cifrado basado en las Secure Socket Layers (SSL) para crear un canal cifrado (cuyo nivel de cifrado depende del servidor remoto y del navegador utilizado por el cliente) más apropiado para el tráfico de información. Es utilizado principalmente por entidades bancarias, tiendas en línea, y cualquier tipo de servicio que requiera el envío de datos personales o contraseñas.

**IMAP**: Acrónimo del inglés Internet Message Access Protocol, es un protocolo de red de acceso a mensajes electrónicos almacenados en un servidor. Mediante IMAP se puede tener acceso al correo electrónico desde cualquier equipo que tenga una conexión a Internet.

**LDAP:** Acrónimo del inglés Lightweight Directory Access Protocol, es un protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red. Fetichista

**Modelo Vista Controlador (MVC)** es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página, el modelo es el Sistema de Gestión de Base de Datos y la Lógica de negocio y el controlador es el responsable de recibir los eventos de entrada desde la vista.

**NNTP:** Acrónimo del inglés Network News Transport Protocol, es una aplicación de Internet que consiste en un protocolo usado para la lectura y publicación de artículos de noticias.

**POP3:** Acrónimo del inglés Post Office Protocol, se utiliza en internet en clientes locales de correo para obtener los mensajes de correo electrónico almacenados en un servidor remoto.

**Reporte**: Se refiere a la información lógica, relevante, y organizada, obtenida a partir de la recuperación de datos incluidos en el sistema.

**RUP:** El Proceso Unificado de Rational (RUP, en inglés Rational Unified Process) es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

**SNMP:** Conocido también como Protocolo Simple de Administración de Red, es un protocolo de la capa de aplicación que facilita el intercambio de información de administración entre dispositivos de red. Es parte de la familia de protocolos TCP/IP

**Software libre:** Es la denominación del software que brinda libertad a los usuarios sobre un producto adquirido y por tanto, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente.

**UML:** (Unified Modeling Language) Lenguaje Unificado de Modelado. Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad, es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software.

**UNIX:** Registrado oficialmente como **UNIX**®, es un sistema operativo portable, multitarea y multiusuario. Existen varias familias del sistema operativo UNIX que han evolucionado de manera independiente a lo largo de los años. Cada familia se distingue no tanto por sus diferencias técnicas como por sus diferencias en propiedad intelectual.

**XML:** (Extensible Markup Language) Lenguaje de Marcas Extensible. Metalenguaje extensible de etiquetas desarrollado por el W3C. Es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.