INSTITUTO SUPERIOR MINERO METALURGICO DE MOA

FACULTAD DE METALURGIA ELECTROMECANICA

INGENIERIA EN INFORMATICA

Rational Unified Process (RUP): Una Aproximación Metodológica.

Trabajo de Diploma presentado en opción al título de Ingeniera en Informática

Autora: Xenia Sánchez Riverón.

Tutores: Ing.Yadira Romero Rodríguez.

Lic. Yiesenia Rosario Ferrer.

Moa, Holguín

2008

Dedicatoria

A ti Madre mía, por ser mi refugio amado, el ser más extraordinario en mi vida…

Agradecimientos

A Dios y la Virgen Maria por guiar mis pasos…

A mi mamá Inalvis por su dedicación, su confianza y sus desvelos…

A mi papá Luís por sus exigencias, por guiarme por el camino de la razón…

A mi sobrino Pedri por tantas muestras de ternura…

A mis hermanos Ángel Luís y Alexis por su apoyo incondicional…

A Nurys por ser mi confianza…

A Amed, por sus cuidados y presencia constante, por colmarme de los pequeños e infinitos detalles que llenan y conducen mi alma, por su amor que es todo cuanto tengo…

A mis abuelos Anicia, Elva e Israel…

A las familias Sánchez-Riverón por su apoyo, su ayuda, sus incontables muestras cariño, a todos ellos para quienes es eterna mi deuda de gratitud…

A Moira por ser simplemente maravillosa…

A mis amigos Idalberto y Yaritza a los que quiero como a mi propia familia…

A mis compañeros y amigos entrañables Roxana, Yodexy, Tony, Maurice, Franklis y Jose…

A mis tutoras Yiesenia y Yadira…

A Virgen por saber guiarnos con entrega y cariño…

A ellas y a todos los profesores del Departamento y del ISMM de quienes he aprendido que el mejor vicio que existe es APRENDER…

Gracias

Xenia Sánchez Riverón

El sueño se hace a manos y sin permisos

arando el porvenir con viejos bueyes…

Silvio Rodríguez.

Resumen

La creciente necesidad de un efectivo proceso de desarrollo de software para guiar a los desarrolladores se hace más predecible. La demanda de software potente y complejo no se corresponde con cómo se desarrolla el software. El problema del software se reduce a la dificultad que afrontan los desarrolladores para coordinar el trabajo de un gran proyecto. Necesitan un método común, un proceso que proporcione una guía para ordenar las actividades de un equipo.

En el presente Trabajo de Diploma se desarrolla, una guía metodológica para la utilización del Proceso Unificado del Rational (RUP) adaptado para grupos de trabajo pequeños. Con este se dota a la universidad de una completa y detallada documentación que facilita la compresión de estudiantes, profesores y desarrolladores de software.

Abstract

The growing necessity of an effective process of software development to guide to the developers becomes more predictable. The demand of potent and complex software doesn't belong together with how the software is developed. The problem of the software decreases to the difficulties that confront the developers to coordinate the work of a great project. They need a common method, a process that provides a guide to order the activities of a team.

In the present Theses of Diploma is developed a methodological guide for the use of Rational Unified Process (RUP) adaptated for their application in reduced groups of work. With this it is endowed to the university of a complete and detailed documentation that facilitates the compression of students, professors and software developers.

Índice

Introducción 1

Capítulo 1. El Proceso Unificado del Racional ¡Error! Marcador no definido.4

1.1 RUP.... 4
1.1.1 Principales características. 4
1.1.2 Antecedentes 5
1.1.3 Principios de desarrollo. 5
1.2 Ciclo de vida. 7
1.2.1 Fases 7
1.2.2 Flujos de trabajo.... 8
1.2.3 Hitos 8
1.2.4 Trabajadores y Actividades 9
1.2.5 Fase de Inicio FT Modelamiento del Negocio 9
1.2.6 Fase de Inicio FT Captura de Requisitos 10
1.2.7 Fase de Elaboracio'n FT Ana'lisis y Diseño 11
1.2.8 Fase de Elaboracio'n FT Implementacio'n 12
1.2.9 Fase de Elaboracio'n FT Prueba 13
1.2.10 Fase de Construccio'n FT Prueba 13
1.2.11 Fase de Transicio'n 13

Capítulo 2. Fase de Inicio 14

2.1 Introducción.... ¡Error! Marcador no definido.
2.2 Flujo de trabajo de Modelamiento del Negocio. 14
2.2.1 Modelo de Caso de Uso del Negocio 15
2.2.2 Actores del Negocio 15
2.2.3 Caso de Uso del Negocio 15
2.2.4 Realizacion Caso de Uso del Negocio…… 16
2.2.5 Descripcion Textual.... 16
2.2.6 Diagrama de Actividades 16
2.2.7 Diagrama de Clases.... 17
2.2.8 Reglas del Negocio. 17
2.2.9 Listado de Riesgos.... 17
2.2.10 Documento Vision. 17
2.3 Flujo de trabajo de Requerimientos. 18
2.3.1 Requerimientos.No Funcionales 19
2.3.2 Requerimientos. Funcionales.... 19
2.3.3 Actores del Sistema 20
2.3.4 Caso de Uso del Sistema.... 20
2.3.5 Diagrama de Caso de Uso del Sistema 21
2.3.6 Descripcion Textual.... 23

2.3.7 Criterios para el empaquetamiento de los Casos de Usos 23
Capítulo 3. Fase de Elaboración 25
3.1 Introducción 25
3.2 Flujo de trabajo de Análisis y Diseño 25
3.2.1 Modelo Conceptual 25
3.2.2 Estrategias para identificar conceptos 26
3.2.3 Modelo de Análisis 27
3.2.4 Identificacion de Cllases 29
3.2.5 Identificacion de Atributos y Relaciones 31
3.2.6 Diagrama de Cllases 31
3.2.7 Organización en Paquetes 32
3.2.8 Diagrama de Colaboracion 33
3.2.9 Diagrama de Secuencia 34
3.2.10 Modelo de Diseño 36
3.2.11 Papel del Diseño en el ciclo de vida del software 36
3.2.12 Mapa de Navegacion 39
3.2.13 Prototipo fisico de interfaz de usuarios 39
3.2.14 Modelo de Despliegue 39
3.2.15 Vistas Arquitectonicas 40
3.2.16 Arquitectura del sistema 41
3.2.17 Estilos Arquitectonicas 41
3.2.18 Patrones 43
3.2.19 Descripcion de la Arquitectura(vista modelo diseño) 47
3.2.20 Descripcion de la Arquitectura(vista modelo despliegue) 48
3.2.21 Realizacion de los casos de usos del diseño 48
3.2.22 Diagramas de Clases 48
3.2.23 Diagramas de Iteracion 50
3.2.24 Diagramas de Colaboracion 51
3.2.25 Diagramas de Secuencia 51
3.2.26 Requisitos de Implementacion 52
3.2.27 Flujo de Sucesos del Diseño 52
3.2.28 Clases del Diseño 52
3.2.29 Subsistema del Diseño 52
3.2.30 Interfaz 53
3.2.31 Tecnicas comunes de modelado 53
3.2.32 Diseño de la Base de Datos 54
3.3 Flujo de trabajo de Implementación 54
3.3.1 Subsistema 55
3.3.2 Componentes 55

3.3.3 Mapeo del Diseño al codigo 56
3.4 Flujo de trabajo de Prueba 57
3.4.1 Ciclo de Vida de Prueba 58
3.4.2 Flujo de trabajo de la disciplina de Prueba 58
3.4.3 Niveles de Prueba 58
3.4.4 Tipos de Prueba 59
3.4.5 Metodos de Prueba 59
3.4.6 Estrategia de Prueba 59
3.4.7 Diseño de casos de Prueba 60
Capítulo 4. Fase de Construcción y Transicion 61
4.1 Introducción 61
4.2 Flujo de trabajo de Requerimientos 61
4.3 Flujo de trabajo de Análisis y Diseño 61
4.4 Flujo de trabajo de Implementación 61
4.5 Flujo de trabajo de Prueba 63
4.6 Fase de Transicion 64
Capítulo 5. Adaptar RUP para proyectos con equipos de trabajo pequeños 65
5.1 Introducción 65
5.2 Utilización de RUP en nuestro territorio 65
5.3 Analisis de las Encuestas 66
5.4 Propuesta de RUP para proyectos con equipos de trabajo pequeños 68
5.4.1 FT Elicitacion de Requisitos 68
5.4.2 FT Analisis y Diseño 69
5.4.3 FT Implementacion 71
5.4.4 FT Prueba 72
5.5 RUP adaptado en diferentes tipos de proyectos de software para grupos de pocos miembros 73
5.5.1 Propuesta General 74
5.5.2 Propuesta para Aplicaciones de Escritorio 74
5.5.3 Propuesta para Aplicaciones Web 74
5.5.4 Propuesta para Aplicaciones Multimedia 75
Capítulo 6. Ejemplificar RUP adaptado en diferentes tipos de proyectos 76
6.1 Introducción 76
6.2 Ejemplo para Aplicaciones de Escritorio 76
6.3 Ejemplo para Aplicaciones Web 104
6.4 Ejemplo para Aplicaciones Multimedia 117
Conclusiones 143
Recomendaciones 144
Referencias Bibliográficas 145
Glosario de Términos 158

Lista de Figuras

Figura1 Diagrama de los Casos de Usos del Negocio…….....157
Figura2. Diagrama de Actividades……..160
Figura 3 Modelo de Objetos…….....,,161
Figura 4 Diagrama de Paquetes…….....164
Figura 5 Diagrama de Caso de Uso del Sistema…….....165
Figura 6 Diagrama de Clases del Análisis…….....172
Figura.7 Diagrama de Clases del Diseño del Paquete<Gestión de Usuarios>……173
Figura 8 Diagrama de Clases del Diseño del Paquete<Acceso a Datos>……174
Figura 9 Diagrama de Clases del Diseño …….....175
Figura 10 Diagrama de Clases para el Caso de Uso <Actualizar Existencia de Medios>…..176
Figura 11 Aplicando Patrón Experto …….....177
Figura 12 …….....177
Figura 13 Aplicando Patrón "Pure Factory" e Indirección…….....178
Figura 14 Diagrama de Clases del Diseño …….....179
Figura 15 Diagrama de Clases Persistentes…….....182
Figura 16 Modelo Lógico de Datos …….....182
Figura 17 Prototipo de Interfaz…….....183
Figura 18 Diagrama de Despliegue …….....184
Figura 19 Diagrama de Componentes del Paquete <Gestiónde Usuarios>.....185
Figura 20 Diagrama de Componentes del paquete <Gestión de Medios>.....186
Figura 21 Diagrama de Clases del Paquete <*Acceso a Datos*>…….....187
Figura 22 Diagrama de Clases del Diseño…….....188
Figura 22 Aplicando Patrón Experto…….....189
Figura 24 Aplicando Patrón "Pure Factory" e Indirección…….....190
Figura 25 Diagrama de Clases Web…….....191
Figura 26 Mapa de Navegación…….....191
Figura 27 Diagrama de Despliegue…….....192
Figura 28 Diagrama de Componentes del paquete <Gestión de Usuarios>……193
Figura 29 Diagrama de Componentes del paquete <Gestión de Medios>…….. 194
Figura 30 Modelo de Dominio…….....195
Figura 31 Diagrama de Caso de Uso de Presentación…….....199
Figura 32 Diagrama de Caso de Uso de Generales…….....200
Figura 33 Diagrama de Caso de Uso de Examen…….....208
Figura 34. Diagrama de Caso de Uso de Galería…….....212
Figura 35 Diagrama de Caso de Uso de Glosario…….....214
Figura 36 Diagrama de Presentación General…….....217
Figura 37 Diagrama de Presentación Galería de Imágenes…….....217
Figura 38 Diagrama de Presentación Galería de Videos…….....218
Figura 39 Diagrama de Presentación Glosario…….....218
Figura 40 Diagrama de Presentación Examen…….....219
Figura 41 Diagrama Jerárquico de las Clases…….....220
Figura 42 Diagrama de Clases del Paquete Presentación…….....221
Figura 43 Diagrama de Clases del Paquete Galería…….....222
Figura 44 Diagrama de Clases del Paquete Examen…….....224
Figura 45 Diagrama de Clases del Paquete Generales…….....227
Figura 46 Diagrama de Clases del Paquete Glosario…….....232
Figura 47 Diagrama de Clases Persistentes…….....234
Figura 48 Modelo Lógico de los Datos…….....235
Figura 49 Diagrama de Paquetes…….....237
Figura 50 Diagrama de Componentes del Paquete BD_Inglés…….....237
Figura 51 Diagrama de Componentes del Paquete Modelo de Implementación…….....238

Lista de Tablas

Tabla 1 Descripción de los Actores del Negocio……..156
Tabla 2 Descripción de los Trabajadores del Negocio…….....…….…….....157
Tabla 3 Descripción del Caso de Uso <Recoger Medios>……..……..…158
Tabla 4 Definición de Actores del Sistema……...……..……..…164
Tabla 5 Descripción del Caso de Uso del Sistema <Autenticar usuarios>.....……...165
Tabla 6 Descripción del Caso de Uso del Sistema <Actualizar Existencia de Medios>……..166
Tabla 7 Descripción del Caso de Uso del Sistema < Actualizar ficha de Medios>.....168
Tabla 8 Descripción del Caso de Uso del Sistema <Cargar aplicación>…..…….....170
Tabla 9 Descripción de la Clase Persistente <CE_Listado de Medios>…..…….....181
Tabla 10 Descripción de la Clase Persistente<CE_ Medios>…….....181
Tabla 11 Descripción de los Actores del Sistema…….…….....198
Tabla 12 Descripción del caso de uso del sistema <Cargar presentación del sistema>……..199
Tabla 13 Descripción del caso de uso del sistema <Controlar audio del sistema>……...201
Tabla 14. Descripción del caso de uso del sistema <Presentar contenido del tópico selecc>... 202
Tabla 15 Descripción del caso de uso del sistema <Controlar navegación del sistema>……...203
Tabla 16 Descripción del caso de uso del sistema <Presentar ayuda del sistema>…….203
Tabla 17 Descripción del caso de uso del sistema <Admitir salida del cliente del sistema>…...204
Tabla 18 Descripción del caso de uso del sistema <Interactuar con medias del sistema>…….205
Tabla 19 Descripción del caso de uso del sistema <Controlar curso de video o audio>……..206
Tabla 20 Descripción del caso de uso del sistema <Controlar las op con video o audio>……..206
Tabla 21 Descripción del caso de uso del sistema <Obtener información de la BD>……..207
Tabla 22 Descripción del caso de uso del sistema <Solicitar información del cliente>……...209
Tabla 23 Descripción del caso de uso del sistema <Seleccionar el examen>……..……210
Tabla 24 Descripción del caso de uso del sistema <Interactuar con ejercicio en curso>……..211
Tabla 25 Descripción del caso de uso del sistema <Interactuar con ejercicio en curso>……...211
Tabla 26 Descripción del caso de uso del sistema <Mostrar medias de la galería>……213
Tabla 27 Descripción del caso de uso del sistema <Mostrar list palabras del glosario>……214
Tabla 28 Descripción del caso de uso del sistema <Mostrar sig de la palabra del glosario>…215
Tabla 29 Descripción de la tabla Glosario……..235
Tabla 30 Descripción de la tabla Preguntas……...236
Tabla 31 Descripción de la tabla Respuestas……..……236

Introducción

La tendencia actual en el software lleva a la construcción de sistemas más grandes y complejos. Debido a las enormes mejoras en rendimiento de hardware los computadores son más potentes; y los usuarios, por tanto, esperan más de ellos. Pero se necesita mucho más que el último modelo de computador para hacer software de gran calidad.

La fortaleza principal se encuentra en profesionales altamente calificados, en los procesos, las mejores prácticas y las métricas establecidas a lo largo del ciclo de mejora continua en la disciplina de desarrollo de software.

El impacto de este en la sociedad y en la cultura continúa siendo profundo. Al mismo tiempo que crece su importancia, la comunidad del software trata continuamente de desarrollar tecnologías que hagan más sencillo, rápido y menos costosa la fabricación de programas de computadoras de alta calidad.

El desarrollo de software se ha convertido actualmente en un proyecto de gran prioridad para empresas e instituciones que desean entrar en el mundo de la automatización.

Para garantizar excelentes resultados, es indispensable la definición de una metodología flexible que permita ajustarse al desafío que presente cada proyecto.

Este trabajo propone una guía para la utilización del Proceso Unificado del Rational (RUP) como metodología de desarrollo; y a la vez hace una propuesta para una correcta aplicación, en grupos reducidos de trabajo.

RUP es un proceso de Ingeniería de Software que provee un enfoque disciplinado para la asignación de tareas y responsabilidades dentro de una organización desarrolladora de software. Su principal objetivo es asegurar la producción de software de alta calidad que satisfaga las necesidades de sus usuarios finales dentro de un presupuesto y tiempo predecibles. Es un marco de trabajo personalizable, el cual puede fácilmente adaptarse a la manera en que trabaja una compañía. Por lo tanto, RUP puede ser adaptada tanto a empresas grandes como pequeñas y puede ser modificado para adecuarse a las diferentes situaciones.

Por la carencia de una documentación referente a esta metodología, que brinde información sobre su eficiente aplicación en los distintos tipos de proyectos, así como la necesidad de adaptar la misma para alcanzar resultados satisfactorios en equipos de pocos integrantes se impone darle solución al siguiente Problema de Investigación: ¿Cómo garantizar la aplicación eficiente del Proceso Unificado del Rational (RUP) en proyectos desarrollados por grupos de trabajo pequeños?

La introducción de los temas previos permite definir explícitamente el Objeto de Estudio: El Proceso Unificado del Rational (RUP) en el desarrollo de proyectos de software.

El Campo de Acción: El Proceso Unificado del Rational (RUP) en el desarrollo de proyectos por equipos de trabajo pequeños.

Se da curso a la investigación entonces con la siguiente Idea a Defender, desarrollar una guía metodológica que permita la comprensión de RUP y su adaptación para el desarrollo de proyectos de software por grupos de trabajo pequeños, lo que permitirá obtener resultados relevantes en el desarrollo de proyectos de software por equipos de trabajo de pocos integrantes.

Teniendo en cuenta los puntos desarrollados hasta aquí, se proponen como objetivos del trabajo:

Objetivo General

Adaptar RUP para su eficiente aplicación por equipos de trabajo pequeños.

Objetivos Específicos

Realizar una revisión bibliográfica para la utilización del Proceso Unificado del Rational (RUP) que incluya las fases y los flujos de trabajo fundamentales, documentación e hitos de cada etapa.

1. Adaptar RUP para proyectos con equipos de trabajo pequeños.
2. Ejemplificar la utilización de RUP adaptado en diferentes tipos de proyectos de software para grupos de pocos miembros.
3. Elaborar un ejemplo utilizando RUP para grupos de pocos desarrolladores.

Tareas

- Realizar una investigación que abarque las fases y flujos de trabajo fundamentales de RUP.
- Efectuar investigaciones de sugerencias para adaptar RUP para su eficiente aplicación en grupos reducidos de trabajo.
- Realizar encuestas a desarrolladores de software, líderes de proyecto y procesar los resultados de las encuestas.
- Analizar y definir los roles, las actividades que estos realizan, y los artefactos para la propuesta de RUP.
- Definir artefactos que se generan según el tipo de aplicación en proyectos con grupos pequeños.
- Aplicar esta propuesta para diferentes tipos de aplicaciones.

Para el Instituto Superior Minero Metalúrgico de Moa Dr. Antonio Núñez Jiménez como centro vinculado a la labor investigativa en nuestro país, y a la inserción de los estudiantes al perfil productivo como uno de los propósitos del proceso docente-educativo, la presente indagación

tendría un gran valor, ya que dotaría a esta entidad de las conocidas como “buenas prácticas” en el progreso de software actual, las cuales se deben tener presentes en el desarrollo de aplicaciones empresariales para garantizar el éxito de los proyectos en las mismas, cuya meta está delimitada en la automatización a corto plazo de estos. Obteniéndose productos más sólidos y confiables.

El presente documento se estructura en 6 capítulos:

Capítulo 1: El Proceso Unificado del Rational, se profundiza en los fundamentos teóricos de este tema y en los aspectos necesarios para su entendimiento.

Capítulo 2: Fase de Inicio, en este capítulo se presenta cómo realizar cada uno de los artefactos que se generan en esta fase.

Capítulo 3: Fase de Elaboración, se describen los artefactos que se obtienen en esta fase y cómo construir cada uno de ellos.

Capítulo 4: Fase de Construcción y Transición, se especifica qué se debe hacer en estas fases.

Capítulo 5: Adaptar RUP para proyectos con equipos de trabajo pequeños, en este capítulo se muestran los resultados obtenidos en las encuestas, y la propuesta de RUP adaptado donde se definen los roles, actividades y artefactos para este tipo de proyectos. También se presenta una propuesta de RUP adaptado según el tipo de aplicación (Escritorio, Web, Multimedia) para grupos de pocos miembros.

Capítulo 6: Ejemplificar RUP adaptado en diferentes tipos de proyectos, se realiza un ejemplo para cada tipo de aplicación, los que ayudarán a la rápida comprensión de la propuesta.

Capítulo 1 El Proceso Unificado del Rational

En un proyecto de desarrollo de software la metodología define quién debe hacer qué, cuándo y cómo debe hacerlo. Una metodología es un proceso. No existe una metodología de software universal; las características de cada proyecto (equipo de desarrollo, recursos, etc.) son las que determinan la más apropiada. En este capítulo se sientan las bases para comprender todos los aspectos relacionados con RUP como metodología para el desarrollo de software.

1.1 RUP

RUP es una metodología sólida, con documentación, que apoya el ciclo de vida evolutivo incremental, además de orientarse al desarrollo de componentes, apoyando el desarrollo orientado a objetos [3].

En el proceso de desarrollo, RUP (Rational Unified Process) aplica varias de las mejores experiencias en el desarrollo moderno de software en una forma que se adapta a un amplio rango de proyectos y organizaciones. Provee a cada miembro del equipo, de un fácil acceso a una base de conocimiento con guías, plantillas y herramientas para todas las actividades críticas del desarrollo de software. Permite que todos los integrantes de un equipo de trabajo, conozcan y compartan el proceso de desarrollo, una base de conocimientos y los distintos modelos de cómo desarrollar el software utilizando un lenguaje de modelado común: UML.

Es un marco de trabajo (Framework) que puede ser adoptado y extendido para satisfacer las necesidades de la organización que lo utilice seleccionando las fases e iteraciones, los flujos de trabajo y disciplinas que se van a recorrer, y los, entregables o productos (artefactos) que se van a construir.

1.1.1 Principales características

El RUP es un producto de Rational (IBM). Se caracteriza por ser:

I. Dirigido por casos de uso: Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso (cómo se llevan a cabo).

II. Centrado en la arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que

son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente. RUP se desarrolla mediante iteraciones, comenzando por los CU relevantes desde el punto de vista de la arquitectura. Tal como se aprecia en la Figura 14, el modelo de arquitectura se representa a través de vistas en las que se incluyen los diagramas de UML.

III. Iterativo e Incremental: Aunque la figura 8 puede sugerir que los flujos de trabajo se desarrollan en cascada, la "lectura" de este gráfico tiene que ser vertical y horizontal. RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Por ejemplo, una iteración de elaboración centra su atención en el análisis y diseño, aunque refina los requerimientos y obtiene un producto con un determinado nivel, pero que irá creciendo incrementalmente en cada iteración.

Además incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso). Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo). Posibilita la administración de requisitos, modelado visual del software, verificación de la calidad en forma continua y control de cambios.

1.1.2 Antecedentes de RUP

El UP es considerado por sus creadores como un proceso completo en el desarrollo de software, respaldado por más de tres décadas de desarrollo y de aplicaciones prácticas. A continuación se presenta una síntesis de la evolución del UP.

El UP tiene sus orígenes en 1967 en el modelo de Ericsson, el cual era similar a un modelo de desarrollo basado en componentes, cuyo creador fue Ivar Jacobson, el mismo que siguió mejorando su método durante los años siguientes, hasta que en 1987 abandona a Ericsson y funda Objectory AB, organización dedicada al desarrollo del proceso llamado Objectory, que significa "fabrica de objetos".

En ese entonces ya existían los lenguajes de descripción y especificación, tal es el caso del SDL (Specification and Description Languages – Lenguaje de Especificación y Descripción) que era considerado como un estándar el modelado de objetos. Este lenguaje fue influenciado por el método de Ericsson, y se utilizó como lenguaje de modelado de Objectory. Objectory creció desde la primera versión en 1988 hasta la versión 3.8 en 1995.

En 1995 Rational Software Corporation compró a Objectory AB, a partir de donde surgió la idea de unificar los procesos de desarrollo. En 1996 surge Objectory de Rational 4.1 que contenía la

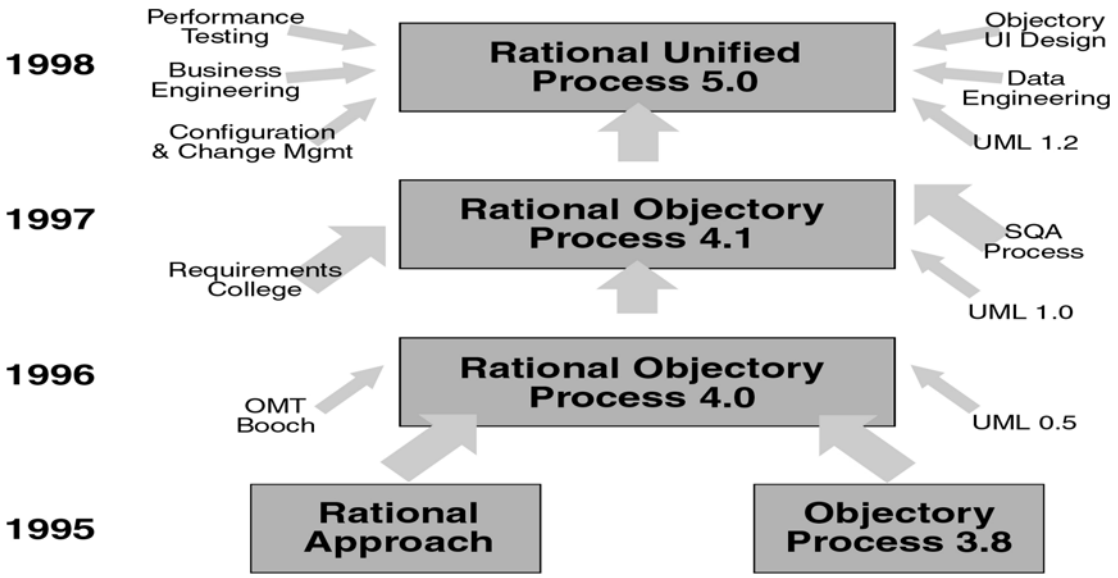
experiencia y práctica de Rational. En ese año UML estaba en fase de desarrollo y se incorporó como el lenguaje de modelado de Objectory 4.1.

A finales de 1997, Rational se fusionó con otras empresas de ingeniería de software en aras de mezclar sus experiencias en el área de los procesos de desarrollo y mejorar a Objectory.

El Proceso Unificado del Rational fue el resultado de una convergencia de Rational Approach y Objectory (el proceso de la empresa Objectory AB). El primer resultado de esta fusión fue el Rational Objectory Process.

En Junio de 1998 Rational publicó el RUP 5.0 (Rational Unified Process - Proceso Unificado de Rational) siendo el arquitecto en jefe Philippe Kruchten. Creado por Jacobson, Rumbaugh y Booch; RUP unifica los mejores elementos de metodologías anteriores.

A partir del surgimiento de este, se tiene un Proceso Unificado para soportar todo el ciclo de vida de un sistema de software, el cual se sigue mejorando con la ayuda de muchas empresas y desarrolladores. Actualmente Rational Software Corporation es parte de IBM.



1.1.3 Principios de desarrollo

El RUP está basado en 6 principios claves:

1. Adaptar el proceso: El proceso deberá adaptarse a las características propias del proyecto u organización. El tamaño del mismo, así como su tipo o las regulaciones que lo condicionen, influirán en su diseño específico. También se deberá tener en cuenta el alcance del proyecto.

2. Balancear prioridades: Los requerimientos de los diversos inversores pueden ser diferentes, contradictorios o disputarse recursos limitados. Debe encontrarse un balance que satisfaga los deseos de todos.

3. Colaboración entre equipos: El desarrollo de software no lo hace una única persona sino múltiples equipos. Debe haber una comunicación fluida para coordinar requerimientos, desarrollo, evaluaciones, planes, resultados, etc.

4. Demostrar valor iterativamente: Los proyectos se entregan, aunque sea de un modo interno, en etapas iteradas. En cada iteración se analiza la opinión de los inversores, la estabilidad y calidad del producto, y se refina la dirección del proyecto así como también los riesgos involucrados.

5. Elevar el nivel de abstracción: Este principio dominante motiva el uso de conceptos reutilizables tales como patrón del software, lenguajes 4GL o esquemas (frameworks) por nombrar algunos. Esto previene a los ingenieros de software ir directamente de los requisitos a la codificación de software, a la medida del cliente. Un nivel alto de abstracción también permite discusiones sobre diversos niveles arquitectónicos. Éstos se pueden acompañar por las representaciones visuales de la arquitectura, por ejemplo con UML.

6. Enfocarse en la calidad: El control de calidad no debe realizarse al final de cada iteración, sino en todos los aspectos de la producción.

Otras características o ventajas de la aplicación de esta metodología son las siguientes:

• Reconoce que las necesidades del usuario y sus requerimientos no se pueden definir completamente al principio.

• Permite evaluar tempranamente los riesgos en lugar de descubrir problemas en la integración final del sistema.

• Reduce el costo del riesgo a los costos de un solo incremento.

• Acelera el ritmo del esfuerzo de desarrollo en su totalidad debido a que los desarrolladores trabajan para obtener resultados claros a corto plazo.

• Distribuye la carga de trabajo a lo largo del tiempo del proyecto ya que todas las disciplinas colaboran en cada iteración.

• Facilita la reutilización del código teniendo en cuenta que se realizan revisiones en las primeras iteraciones lo cual además permite que se aprecien oportunidades de mejoras en el diseño.

1.2 Ciclo de vida

Un típico perfil de proyecto mostrando el tamaño relativo de las cuatro fases. El ciclo de vida RUP es una implementación del Desarrollo en espiral. Fue creado ensamblando los elementos en secuencias semi-ordenadas. El ciclo de vida organiza las tareas en fases e iteraciones.RUP divide el proceso de desarrollo en ciclos, teniendo un producto final al final de cada ciclo, cada ciclo se divide en fases que finalizan con un hito donde se debe tomar una decisión importante.

1.2.1 Fases

Las fases dividen el proceso de desarrollo a lo largo del tiempo, cada una de las cuales tiene objetivos específicos y un conjunto de "artefactos" definidos que deben alcanzarse. La duración de cada fase depende del equipo y del producto a generar.

A su vez, cada fase puede tener una o más iteraciones y cada iteración sigue el modelo en cascada pasando por las distintas disciplinas. Cada iteración termina con una liberación del producto.

- Establece oportunidad y alcance.
- Identifica las entidades externas o actores con las que se trata.
- Identifica los casos de uso.

Conceptualización (Concepción o Inicio): Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema y se identifican los riesgos.

Elaboración: Se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos. Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen. A pesar de que se desarrolla a profundidad una parte del sistema, las decisiones sobre la arquitectura se hacen sobre la base de la comprensión del sistema completo y los requerimientos (funcionales y no funcionales) identificados de acuerdo al alcance definido.

Construcción: Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario. Se obtiene 1 o varios release del producto que han pasado las pruebas. Se ponen estos release a consideración de un subconjunto de usuarios.

Transición: El release ya está listo para su instalación en las condiciones reales. Se instala el producto al cliente y se entrena a los usuarios. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados. Lo que puede implicar reparación de errores.

Es práctico dividir el trabajo en partes más pequeñas o miniproyectos. Cada miniproyecto es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en los flujos de

trabajo, y los incrementos, al crecimiento del producto. Cada iteración se realiza de forma planificada es por eso que se dice que son miniproyectos.

1.2.2 Hitos

Cada fase finaliza con un hito a continuación se presentan los hitos de las fases de RUP:

Conceptualización	Objetivos
Elaboración	Arquitectura
Construcción	Funcionalidad operativa
Transición	Release del sistema

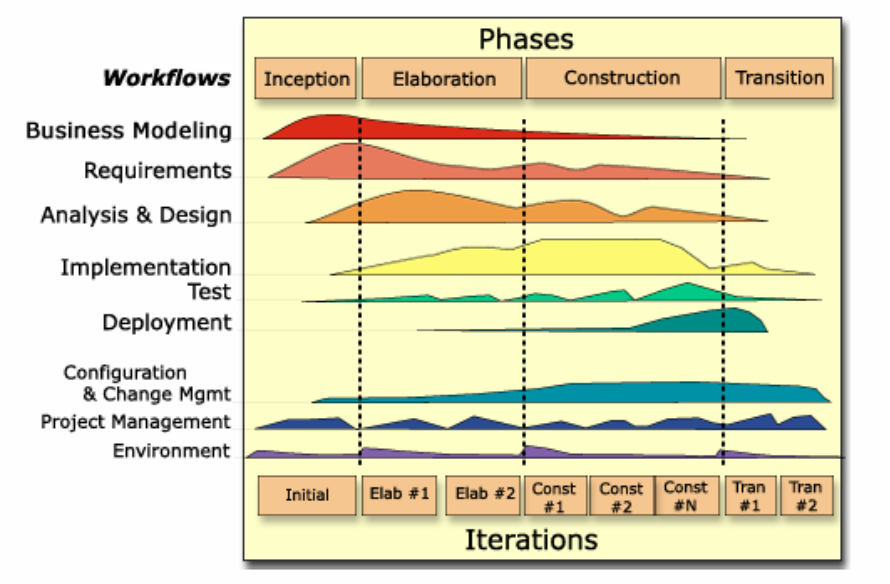
1.2.3 Flujos de Trabajo (FT)

Los Flujos de Trabajo (Cuándo) son una secuencia de actividades realizadas por los trabajadores que producen un resultado de valor observable. Estas actividades están ordenadas, así que una actividad produce una salida que sirve de entrada a la siguiente actividad.

En RUP se han agrupado las actividades en grupos lógicos definiéndose 9 flujos de trabajo principales. Los 6 primeros son conocidos como flujos de ingeniería y los tres últimos como de apoyo.

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

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



1.2.4 Trabajadores y Actividades

El Proceso Unificado del Rational especifica quién es el encargado de hacer cada cosa, con la definición de los trabajadores, y estos realizan cada una de las actividades.

Trabajador ("quién"): Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo.

Actividades ("cómo"): Es una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos.

1.2.6 Fase de Inicio. FT Modelamiento del Negocio

El Flujo de Trabajo de Modelamiento del Negocio da una visión de qué es necesario hacer para dar respuesta a las solicitudes del usuario, lo cual se logra definiendo los procesos, roles y responsabilidades de la organización en los modelos de casos de uso del negocio y de objetos.

Los trabajadores que participan en este flujo de trabajo son:

Analista de procesos del negocio: Responsable de la arquitectura del negocio por lo que dirige y coordina el proceso de modelamiento del negocio. Decide cuáles son los actores y los procesos del negocio y la relaciones entre ellos y cuáles son las reglas del negocio a tener en cuenta.

Diseñador del negocio: Describe los procesos del negocio y como parte de la realización de estos procesos identifica a la entidades y trabajadores del negocio y sus relaciones. Define cuáles son los requerimientos en la automatización.

Revisor del modelo del negocio (arquitecto): Revisa formalmente el modelo de CUN y de objetos del negocio.

1.2.5 Fase de Inicio. FT Captura de Requisitos

En este flujo de trabajo se define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.

Los trabajadores que participan en este flujo de trabajo son:

Analista del sistema: Define los alcances del sistema e identifica a los actores y casos de uso que permiten modelar completa y consistentemente el sistema.

- Desarrolla su trabajo fundamentalmente en los primeros flujos de trabajo.
- Debe ser un experto identificando problemas y oportunidades, teniendo además la habilidad de asociar a los problemas la solución y de determinar la oportunidad de resolverlo.
- Identifica las demandas de los stakeholders[1].
- Desarrolla el Plan de Gestión de Requerimientos.
- Desarrolla el Documento Visión.
- Captura los requisitos del sistema (funcionales, de usabilidad, de capacidad, etc.)
- Identifica Actores y Casos de Uso del sistema.
- Estructurar el modelo de Casos de Uso del sistema

Especificador de casos de uso: Describe detalladamente cada caso de uso de acuerdo a las funcionalidades que engloba.

Diseñador de interfaz de usuario: Responsable de desarrollar el prototipo de la interfaz de usuario para algunos casos de uso, habitualmente un prototipo para cada actor.

Arquitecto: Describe la vista de la arquitectura del modelo de casos de uso, definiendo la prioridad de cada caso de uso para decidir en qué iteración será desarrollado cada uno.

[1] Ver Glosario de Términos

1.2.8 Fase de Elaboración. FT Análisis y Diseño

El objetivo de este flujo de trabajo es traducir los requisitos a una especificación que describe cómo implementar el sistema.

FT Análisis

Los trabajadores que participan son:

- Arquitecto: Responde por la integridad del modelo de análisis y por la arquitectura del modelo de análisis.
- Ingeniero de casos de uso: Responde por la integridad de una o más realizaciones de casos de uso (descripción textual del flujo de sucesos, Diagrama de clases que muestra las clases del análisis participantes y los diagramas de interacción que muestran la realización de un flujo particular del caso de uso en términos de objetos del análisis), así como de que respondan los casos de uso a los requisitos que engloba cada uno.
- Ingeniero de componentes: Define y mantiene las clases y las relaciones entre ellas y la integridad de uno o varios paquetes del análisis.

FT Diseño

El diseño es el centro de atención al final de la fase de elaboración y el comienzo de las iteraciones de construcción. Esto contribuye a una arquitectura estable y sólida, y crear un plano del modelo de implementación.

Los trabajadores que participan son:

Arquitecto: Responsable del modelo de diseño, modelo de despliegue, descripción de la arquitectura.

Ingeniero de casos de uso: Responsable de realización de caso de uso-diseño.

Ingeniero de componentes: Responsable de clase del diseño, subsistema de diseño, interfaz.

Diseño de Base de Datos

Trabajador: Diseñador de BD

Debe tener conocimientos sólidos sobre:

- Modelado de datos.
- Diseño de bases de datos.
- Técnicas de análisis y diseño orientado a objetos.
- Arquitectura de sistemas.
- Administración de bases de datos.

1.2.9 Fase de Elaboración. FT Implementación

El Flujo de Trabajo de Implementación describe cómo los elementos del Modelo del Diseño se implementan en términos de componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación. Creando un software que se ajuste a la arquitectura y que tenga el comportamiento deseado.

Trabajadores y Actividades

Trabajadores	Principales Actividades
Arquitecto	Estructurar el Modelo de Implementación.
Integrador	Planificar la Integración al Sistema
Programador	Implementación de Componentes

1.2.10 Fase de Construcción. FT Implementación

La implementación es el centro durante las iteraciones de construcción, aunque también se lleva a cabo el trabajo de implementación durante la fase de elaboración, para crear la línea base ejecutable de la arquitectura, y durante la fase de transición, para tratar defectos tardíos como los encontrados con distribuciones beta del sistema.

Trabajadores

- Arquitecto: Responsable de la integridad, corrección, completitud y legibilidad del modelo de implementación de acuerdo a lo descrito en el modelo de diseño. Es responsable también de la arquitectura del modelo de implementación, es decir, de la existencia de sus partes significativas para arquitectónicamente. Un resultado importante de la implementación es la asignación de componentes ejecutables a nodos. El arquitecto es responsable de esta asignación. Como estamos dentro de la Fase de Construcción, la tarea del Arquitecto se reduce bastante, ya que se supone que la arquitectura está creada y es estable, este solo se limitaría de actualizarla en caso de cambios.
- Ingeniero de componentes: Define y mantiene uno o varios componentes y subsistemas, garantizando que cada componente implemente la funcionalidad correcta. También mantiene la integridad de uno o varios subsistemas de implementación. Debe garantizar que los elementos de los subsistemas de implementación son correctos, que sus dependencias con otros subsistemas o interfaces son correctas y que implementan correctamente las interfaces que proporcionan. Es natural que un ingeniero de componentes tenga bajo su responsabilidad un subsistema y los

elementos que lo componen durante las etapas de diseño e implementación, pero también este es responsable de realizarles pruebas de unidad a los componentes que crea.

- Integrador de sistemas: Planifica la secuencia de construcciones necesarias en cada iteración y la integración de cada construcción cuando sus partes han sido implementadas. Esta planificación da lugar a un plan de integración de construcciones.

1.2.11 Fase de Elaboración. FT Prueba

En este Flujo de Trabajo el sistema o componente es ejecutado bajo unas condiciones o requerimientos específicos, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente.

Trabajadores

- Administrador de Prueba: Es el responsable del éxito de la prueba, Este rol involucra defensor de prueba y calidad, planificación y administración de recursos, resolución de problemas que impiden las pruebas.
- Analista de Prueba: Es el responsable de identificar y definir las pruebas requeridas, monitorear el progreso de la prueba y el resultado en cada ciclo de prueba y evaluando la calidad total experimentada como un resultado de las actividades de prueba. Este rol lleva la responsabilidad para representar apropiadamente las necesidades de los stakeholders que no tienen representación regular y directa en el proyecto.
- Diseñador de prueba: Es el responsable de definir el método de prueba y asegurar su implementación exitosa. El rol incluye identificando técnicas apropiadas, herramientas e instrucciones para implementar las pruebas necesarias y encauzar los recursos correspondientes para las pruebas.
- Probador: Es el responsable de las actividades principales de las pruebas, el cual incluye la conducción de las pruebas necesarias y el registro del resultado de la prueba.

1.2.12 Fase Construcción. FT Prueba

En esta fase y flujo de trabajo se realizan las mismas actividades e intervienen los mismos trabajadores que en la fase de Elaboración.

1.2.13 Fase de Transición

En esta fase el release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores. En la misma intervienen el usuario final y el equipo de trabajo.

Capítulo 2 Fase de Inicio

2.1 Introducción

Con el desarrollo del presente capítulo se describe la Fase de Inicio de RUP con los flujos de trabajo que se desarrollan en el mismo, así como las actividades y los artefactos que se generan en cada uno de ellos; se recomienda leer [7], [15] y otros libros y documentos que tratan el tema con mayor profundidad.

2.2 Flujo de Trabajo Modelamiento del Negocio

El proceso de modelamiento permite obtener una visión de la organización que permita definir los procesos, roles y responsabilidades; en los modelos de casos de uso del negocio y de objetos.

Este proceso está relacionado con los de obtención de requerimientos y análisis-diseño. En la primera iteración se hará una evaluación de la organización en la cual se implantará el futuro sistema y, basado en los resultados, se tomarán decisiones sobre cómo continuará esa iteración. Dependiendo de la situación o escenario que se presente, hay varias alternativas de desarrollar este proceso:

Si se determina que no es necesario un modelo completo del negocio se realizará lo que se conoce como un modelamiento del dominio. [10]

Un modelo del dominio captura los tipos más importantes de objetos que existen o los eventos que suceden en el entorno donde estará el sistema.

El modelo del dominio se considera en RUP un subconjunto del llamado modelo de objetos del negocio. Esos modelos, que no incluyen las responsabilidades de las personas que ejecutan las actividades, se refieren a veces como modelo del dominio.

Si se determina que no habrán cambios importantes en los procesos de negocio, entonces solo es necesario modelar el negocio propuesto.

La descripción del negocio propuesto en detalle tendrá entre sus actividades principales la identificación de los procesos de negocio, delimitación el modelo de casos de uso del negocio, la especificación de los casos de uso del negocio, la identificación de trabajadores y entidades del negocio que ejecutan las realizaciones de los casos de uso del negocio y detallar la definición de las entidades del negocio y las responsabilidades de los trabajadores del negocio.

Artefactos (”qué”): Productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.

Los artefactos que se generan en este flujo de trabajo son:

- Descripción del Negocio2
- Actor
- Caso de uso
- Realización de cada CUN
- Diagramas de actividades
- Modelo de Objeto
- Glosario de términos
- Reglas del Negocio
- Listado de riesgos
- Documento Visión

2.2.1 Modelo de Casos de Uso del Negocio

El Modelo de Casos de Uso del Negocio[3] describe los procesos de un negocio (casos de uso del negocio) y su interacción con elementos externos (actores), tales como socios y clientes, es decir, describe las funciones que el negocio pretende realizar y su objetivo básico es describir cómo el negocio es utilizado por sus clientes y socios.

2.2.2 Actores del Negocio

Un actor del negocio es cualquier individuo, grupo, entidad, organización, máquina o sistema de información externos; con los que el negocio interactúa. Lo que se modela como actor es el rol que se juega cuando se interactúa con el negocio para beneficiarse de sus resultados.

Para cada actor del negocio que se identifica se debe escribir una breve descripción que incluya sus responsabilidades y por qué interactúa con el negocio.

Un actor del negocio representa un tipo particular de usuario del negocio más que un usuario físico, ya que varios usuarios físicos pueden realizar el mismo papel en relación al negocio, o sea, ser instancias de un mismo actor. De otro lado un mismo usuario puede actuar como diferentes actores.

2.2.3 Casos de Uso del Negocio (CUN)

Un caso de uso del negocio[4] representa a un proceso de negocio, por lo que se corresponde con una secuencia de acciones que producen un resultado observable para ciertos actores del negocio. Desde la perspectiva de un actor individual, define un flujo de trabajo completo que produce resultados deseables.

[2] Ver anexo 3
[3] Ver estereotipos usados en el Modelo de Casos de Uso del Negocio en Anexo 2
[4] Ver ejemplo de Diagrama de Casos de Uso del Negocio en Capítulo 6

Se pueden agrupar casos de uso[5] del negocio para particionar el diagrama en subdiagramas más pequeños; de manera que se definirían paquetes y estos a su vez podrían relacionarse entre sí. Un paquete (estereotipo: 🗀) es un mecanismo de propósito general para organizar en grupos los elementos.

Para la elaboración de los diagramas se recomienda revisar [10] y otros documentos que traten el tema con mayor profundidad.

2.2.4 Realización de los Casos de Uso del Negocio

La realización de un caso de uso de negocio muestra cómo colaboran los trabajadores y entidades de negocio para ejecutar el proceso. Cada realización se puede documentar y utilizando los diagramas de actividades, secuencia y clases y descripciones textuales. [10] Consideramos que con una descripción textual, el diagrama de actividades y las clases, es suficiente para describir completamente el proceso de negocio y dan información necesaria para los flujos de trabajo que se ejecutan posteriormente.

Un trabajador del negocio es una abstracción de una persona (o grupo de personas), una máquina o un sistema automatizado; que actúa en el negocio realizando una o varias actividades, interactuando con otros trabajadores del negocio y manipulando entidades del negocio. Representa un rol.

Las entidades de negocio representan a los objetos que los trabajadores del negocio toman, inspeccionan, manipulan, producen o utilizan durante la realización de los casos de uso de negocio. Comúnmente representan un documento o una parte esencial de un producto. Algunas veces representa cosas no tangibles como el conocimiento acerca de un mercado o cliente.

2.2.5 Descripción textual

La descripción textual de un caso de uso de negocio se formaliza en un documento generalmente llamado Especificación del caso de uso de negocio[6].

2.2.6 Diagrama de Actividades

El diagrama de actividad[7] es un grafo (grafo de actividades) que contiene estados en que puede hallarse una actividad. Un estado de actividad representa la ejecución de una sentencia de un procedimiento, o el funcionamiento de una actividad en un flujo de trabajo. En vez de esperar un

[5] Ver criterios para agrupar caso de uso en anexo 4
[6] Ver formato en anexo 5
[7] Ver como crearlos en anexo 6

evento, como en un estado de espera normal, un estado de actividad espera la terminación de su cómputo. Cuando la actividad termina, entonces la ejecución procede al siguiente estado de actividad dentro del grafo. Una transición de terminación es activada en un diagrama de actividades cuando se completa la actividad precedente. [10]

2.2.7 Diagrama de Clases

El diagrama de clases, como artefacto que se construye para describir el modelo de objetos del negocio, describe cómo colaboran los trabajadores y entidades del negocio dentro del flujo de trabajo del proceso del negocio. [10]

2.2.8 Reglas del Negocio

Las reglas de negocio[8] describen políticas que deben cumplirse o condiciones que deben satisfacerse, por lo que regulan algún aspecto del negocio.

2.2.9 Listado de Riesgos

La administración del riesgo consiste en ocuparse de las incógnitas de un proyecto, las cuestiones que pueden llevarlo a pique. En concreto hay que identificar los riesgos, para tratar de evitarlos, transferirlos o asumirlos. En este último caso habrá que tratar de mitigar el riesgo y definir un plan de contingencia por si el riesgo se convierte en un problema real. Monitorizar un proyecto es importante para mantenerlo bajo control. Se tiene que "medir" para ver cuanto se ajusta a los planes, la calidad requerida y los requisitos. También es necesario para planificar de forma precisa y ver cuál es el comportamiento del proyecto frente a cambios.

2.2.10 Documento Visión

- Representa un documento importante en el hito de la fase de Inicio.
- Es uno de los artefactos esenciales.
- Tiene plantilla.
- Una de las secciones de la plantilla es: Descripción de los usuarios y todos los involucrados.
- No se usan estereotipos de UML en el artefacto.
- Es creado por el Analista de procesos del negocio.
- Es creado al principio de la fase de inicio, evoluciona en las porciones más temprana del ciclo de vida.

[8] Ver clasificación en anexo 7

- Uno de los errores más comunes en la realización del artefacto es que no se describe correctamente el problema que se trata en el proyecto.
- Una de la secciones de la plantilla es Principales necesidades de los involucrados y los usuarios
- Se crea en el flujo de trabajo Modelación del Negocio.

2.3 Flujo de trabajo de Requerimientos

La fase de Inicio tiene por finalidad definir la visión, los objetivos y el alcance del proyecto, tanto desde el punto de vista funcional como del técnico, obteniéndose como uno de los principales resultados una lista de los Casos de Uso y una lista de los factores de riesgo del proyecto. Como ya se ha mencionado en clases anteriores el principal esfuerzo se lleva a cabo en los dos primeros flujos de trabajo en el "Modelamiento del Negocio" y el "Análisis de Requerimientos".

Como parte de este flujo de trabajo las principales actividades que se realizan son:

- Identificar y clasificar requerimientos.
- Encontrar actores y casos de uso.
- Priorizar casos de uso.
- Detallar casos de uso.
- Prototipar la interfaz de usuario.
- Estructurar el modelo de casos de uso.

Los artefactos de este flujo de trabajo son:

- Modelo de casos de uso: Es un modelo del sistema que contiene actores, casos de uso y sus relaciones.
- Actor: Terceros fuera del sistema que interactúan con él.
- Caso de uso: Fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores.
- Descripción de la arquitectura (vista del modelo de casos de uso): Representa los casos de uso significativos para la arquitectura ya que describen alguna funcionalidad importante y crítica o algún requisito que deba priorizarse.
- Glosario de términos: Términos comunes que se utilizan para describir el sistema.
- Prototipo de interfaz usuario: Presentación de la interfaz del producto que representa la funcionalidad contenida en los casos de uso; de manera que permita que el usuario verifique que el sistema va a satisfacer sus necesidades.

Muchas aplicaciones fallan porque existen incongruencias entre lo que el usuario quería, lo que realmente necesitaba, lo que interpretaba cada miembro del equipo de proyecto y lo que realmente se obtiene. Es por ello que las áreas de esfuerzo del análisis de requerimientos están en:

- Reconocimiento del problema como lo ve el usuario.
- Evaluación del problema y síntesis de la evaluación
- Modelado: Creación de modelos que ayuden a entender la entidad a construir.
- Construcción de un prototipo de alto nivel del sistema.
- Revisión por parte del usuario.
- Firma del contrato si las partes están de acuerdo.

2.3.1 Requerimientos no Funcionales

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

Existen múltiples categorías para clasificar a los requerimientos no funcionales, siendo las siguientes representativas de un conjunto de aspectos que se deben tener en cuenta, aunque no limitan a la definición de otros.

- Requerimientos de Software
- Requerimientos de Hardware
- Restricciones en el Diseño y la Implementación
- Requerimientos de apariencia o interfaz externa
- Requerimientos de Seguridad
- Requerimientos de Usabilidad
- Requerimientos de Soporte
- También se tienen:
- Requerimientos Legales
- Requerimientos de Confiabilidad
- Requerimientos de interfaz Interna

2.3.2 Requerimientos Funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir.

En la realización de los casos de uso del negocio, se obtienen las actividades que serán objeto de automatización. Estas actividades no son exactamente los requerimientos funcionales, pero si son el punto de partida para identificar qué debe hacer el sistema.

Los requerimientos funcionales no alteran la funcionalidad del producto, esto quiere decir que los requerimientos funcionales se mantienen invariables sin importarle con que propiedades o cualidades se relacionen.

Los requerimientos deben ser especificados por escrito, como un acuerdo entre las partes y descritos como una característica del sistema a entregar.

2.3.3 Actores del Sistema

Cada trabajador del negocio (inclusive si fuera un sistema ya existente) que tiene actividades a automatizar es un candidato a actor del sistema. Si algún actor del negocio va a interactuar con el sistema, entonces también será un actor del sistema.

Los actores del sistema:

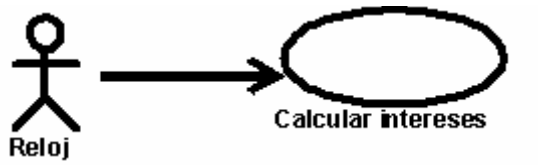
- No son parte de él.
- Pueden intercambiar información con él.
- Pueden ser un recipiente pasivo de información.
- Pueden representar el rol que juega una o varias personas, un equipo o un sistema automatizado.

2.3.4 Casos de Uso del Sistema (CUS)

Los casos de uso son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Por lo tanto, establece un acuerdo entre clientes y desarrolladores sobre las condiciones y posibilidades (requisitos) que debe cumplir el sistema.

Los casos de uso candidatos también se encuentran entre las actividades a automatizar. Esto no significa que una actividad se convierta en un caso de uso porque un caso de uso es un proceso que da un resultado de valor para un actor determinado y una secuencia de actividades a automatizar puede implicar pasos dentro de un caso de uso.

En algunos sistemas se tienen actividades que se ejecutan periódicamente cuando llega cierto instante de tiempo, como por ejemplo, el cálculo de intereses de los clientes en los bancos se realiza todas las noches. Para modelar esto se define un actor ficticio (Reloj) que es el que “dispara” la ejecución del caso de uso.



La definición de los casos de uso se puede perfeccionar partiendo de que se duplique completamente parte del comportamiento con otros casos de uso o cuando un caso de uso es complejo y largo y su separación facilita que sean manejables y comprensibles. La solución es crear casos de uso independientes definiendo relaciones de inclusión, extensión y generalización / especialización. Esto implica que hay que rescribir el flujo de trabajo de los casos de uso.

¿Qué incluir en una descripción de CU?

1. Definir el estado inicial como precondición.
2. Como y cuando comienza el CU.
3. El orden requerido (si hay alguno) en el que las acciones se deben ejecutar.
4. Como y cuando terminan los CU.
5. Definir los posibles estados finales como poscondiciones.
6. Los caminos de ejecución no están permitidos.
7. Las descripciones de caminos alternativos que están incluidos junto con la descripción del camino básico.
8. Las descripciones de los caminos alternativos que han sido extraídos de la descripción del camino básico.
9. La interacción del sistema con los actores y que cambios producen, o en otras palabras, describimos la secuencia de acciones del CU, como estas acciones son invocadas por los actores y como su ejecución resulta en solicitudes a los actores.
10. La utilización de objetos, valores y recursos de sistema, o dicho de otra forma, describir la secuencia de acciones en la utilización de un CU y asignar valores a los atributos de un CU.
11. Describir explícitamente que hace el sistema, separar las responsabilidades del sistema y la de los actores.

Los atributos de los CU pueden utilizarse mas tarde para encontrar clases y atributos en el análisis y diseño. Los requisitos no funcionales (velocidad, disponibilidad, exactitud, tiempo de respuesta, utilización de memoria) pueden describirse en una sección separada dentro de la descripción de CU.

2.3.5 Diagrama de Casos de Uso del Sistema

Un diagrama de casos de uso del sistema representa gráficamente a los procesos y su interacción con los actores.

Cada caso de uso debe comunicarse con al menos un actor, si no aparece ningún actor que se comunique con un caso de uso esto indica error en el modelo de caso de uso o en los requerimientos planteados. Como excepciones a esta última regla podrían considerarse:

- Si el caso de uso es abstracto (no instanciable) no tiene porque incluir relación con actores (aunque tenerlas)
- Un caso de uso Padre en una relación de generalización-especialización no tiene porque tener relación con un actor si el caso de uso Hijo describe completamente toda la relación con el actor
- Algunos casos de uso se inician acorde a un horario (ejemplo: una vez a la semana o al día) lo que significa que el reloj del sistema es su iniciador. El reloj es interno al sistema, de esta forma el caso de uso no tendría relación de iniciación con ningún actor, pero para esclarecer el modelo debe colocarse un actor ficticio denominado "Reloj".
- Las relaciones de comunicación entre actores y casos de uso no tienen nombre, sólo se necesita especificar el punto de inicio y el fin de la relación. Esto se hace a través de una línea con saeta que indica inicio y fin de la relación.

Un actor se comunica con un caso de uso por múltiples razones:

1. Para invocar un caso de uso. Una instancia de un actor siempre invoca una instancia de un caso de uso.
2. Para solicitar algún dato almacenado en el sistema, el cual el caso de uso obtiene y presenta al actor.
3. Para cambiar el dato almacenado en el sistema mediante el uso de un diálogo con el sistema.
4. Para reportar que algo especial ha ocurrido alrededor del sistema y que dicho sistema debe cuidar.

Un actor inicia un caso de uso. Sin embargo, una vez que este ha comenzado, el caso de uso puede comunicarse con varios actores. Se pueden usar relaciones de comunicación entre casos de uso y actores para mostrar cuáles actores se comunican con ellos. La multiplicidad de las relaciones muestra cuántas instancias del actor se relacionan con una instancia del caso de uso al mismo tiempo.

Los casos de uso de comunican con los actores por muchas razones:

1. Si algo especial ha ocurrido en el sistema, un actor puede necesitar conocerlo.

2. Un caso de uso puede necesitar pedirle a un actor que lo ayude a tomar una decisión si se tienen varias opciones.

3. Es común pero no siempre válido, que el caso de uso espera por una respuesta cuando este le envía una señal al actor. Esto debe quedar explícitamente descrito en el caso de uso.

Estas convenciones pueden esclarecer qué actor inicia el caso de uso y serán las usadas en este curso:

- La flecha de iniciación del actor-caso de uso siempre se muestra, incluso si el caso de uso más tarde inicia comunicación con el actor que lo inició. Esto debe mostrarse sólo con una flecha actor-caso de uso.

- Una flecha caso de uso-actor puede ser omitida o incluida con el objetivo de esclarecer el diagrama.

2.3.6 Descripción Textual

Para entender la funcionalidad asociada a cada caso de uso no es suficiente con la representación gráfica del Diagrama de Casos de Uso. La descripción[9] puede ser elaborada de forma breve o extendida y debe ir acompañada del prototipo respectivo.

2.3.7 Criterios para el empaquetamiento de los CU

Los CU pueden ser asignados a un paquete en concreto. Estos paquetes permiten localizar los cambios en un proceso del negocio, en el comportamiento de un actor y en un conjunto de CU estrechamente relacionados. Por tanto los tres criterios podrían quedar definidos de la siguiente forma:

1. Los CU requeridos para dar soporte a un determinado proceso de negocio.
2. Los CU requeridos para dar soporte a un determinado actor del sistema.
3. Los CU que están relacionados mediante relaciones de generalización y extensión. Este tipo de conjunto de CU es coherente en el sentido de que los CU o bien especializan o bien "extienden" a los otros.

Plantillas existentes:

1. Plan de Gestión de requisitos.
2. Doc. Visión.
3. Doc. Especificación requerimientos.

[9] Formato en anexo 8

4. Modelo de Casos de Uso del Sistema.

Consejos

- Todos los requerimientos funcionales son mostrados en al menos un caso de uso.
- Todos los requerimientos no funcionales que necesitan ser satisfechos por casos de usos específicos tienen que estar mostrados en estos casos de uso.
- En cuanto al modelo de casos de uso recomienda que verifiquemos:
- El modelo debe presentar claramente el comportamiento del sistema; debe resultar fácil de comprender qué hace el sistema revisando el modelo.
- No deben existir largas cadenas de relaciones include y extend ni para los casos de uso extendido ni para los incluidos, esto dificulta la comprensión del diagrama.
- Deben existir una mínima dependencia cuando un caso de uso especializado, incluido o extendido necesita conocer sobre la estructura y contenido de otros casos de uso especializado, incluido o extendido.
- El modelo de casos de uso no debe contener comportamiento superfluo, sino que todos los casos de uso deben quedar justificados al trazar los requerimientos funcionales.
- Todas las relaciones entre casos de uso son requeridas (existe justificación para todas las relaciones include, extend y generalización-especialización).
- Cuando el modelo es grande y/o las responsabilidades del modelo son distribuidas por partes resulta apropiado utilizar paquetes.

Capitulo 3 Fase de Elaboración

3.1 Introducción

En este capítulo se presentan las actividades y los artefactos que se desarrollan en los flujos de trabajo de la Fase de Elaboración de RUP; se recomienda consultar [7], [15] y otros libros.

3.2 Flujo de trabajo de Análisis y Diseño

El objetivo de este flujo de trabajo es traducir los requisitos a una especificación que describe cómo implementar el sistema. El análisis consiste en obtener una visión del sistema que se preocupa de ver QUÉ hace, de modo que sólo se interesa por los requisitos funcionales. Por otro lado, el diseño es un refinamiento del análisis que tiene en cuenta los requisitos no funcionales, en definitiva CÓMO cumple el sistema sus objetivos. El diseño debe ser suficiente para que el sistema pueda ser implementado sin ambigüedades. De hecho, cuando la precisión del diseño es muy grande, la implementación puede ser hecha por un generador automático de código.

Al final de la fase de inicio hay que definir una arquitectura candidata: crear un esquema inicial de la arquitectura del sistema, identificar clases de análisis y actualizar las realizaciones de los casos de uso con las interacciones de las clases de análisis. Durante la fase de elaboración se va refinando esta arquitectura hasta llegar a su forma definitiva. En esta fase se desarrolla el proceso por iteraciones, en cada iteración hay que analizar el comportamiento para diseñar componentes. Además si el sistema usará una base de datos, habrá que diseñarla también, obteniendo un modelo de datos. El resultado final más importante de este flujo de trabajo será el modelo de diseño. Consiste en colaboraciones de clases, que pueden ser agregadas en paquetes y subsistemas. Otro producto importante de este flujo es la documentación de la arquitectura software, que captura varias visiones arquitectónicas del sistema.

3.2.1 Modelo Conceptual

Un paso esencial de un análisis orientado a objetos es descomponer el problema en conceptos u objetos individuales. Un modelo conceptual es una representación de conceptos en un dominio del problema. Además este modelo nos muestra asociaciones entre conceptos y atributos de conceptos.

En UML ilustramos el modelo conceptual como un diagrama de estructura estática, donde no se define ninguna operación.

Además de descomponer el espacio del problema en unidades comprensibles (conceptos), la creación este modelo contribuye a esclarecer la terminología o nomenclatura del dominio. Podemos

verlo como un modelo que comunica cuales son los términos importantes y como se relacionan entre si.

Se puede definir un concepto a partir de su símbolo, su intención, y su extensión.

- Símbolo: Palabra que representa al concepto.
- Intensión: La definición del concepto.
- Extensión: El conjunto de ejemplos a que se aplica el concepto.

Es importante comprender que un modelo conceptual no es una descripción del diseño del software, como una clase de java o C++. Por ello los siguientes elementos no son adecuados en el:

- Los artefactos de software (base de datos, ventanas).
- Las responsabilidades o métodos.

3.2.2 Estrategias para identificar conceptos

Es mejor exagerar y especificar un modelo conceptual con muchos conceptos refinados que no especificarlos cabalmente. Es un error pensar que el modelo conceptual es más adecuado si tiene menos conceptos, pues por lo general suele suceder lo contrario.

No se excluya un concepto simplemente porque los requerimientos no indiquen la no indiquen la necesidad de recordar la información acerca del mismo, o porque el concepto carezca de atributos, es perfectamente válido tener conceptos sin atributos o conceptos con un papel puramente de comportamiento.

Para la obtención de estos conceptos es muy útil usar una lista de categorías de conceptos como la que sigue:

1. Objetos físicos o tangibles
2. Especificaciones, diseño, o descripciones de cosas
3. Lugares
4. Papel de las personas (roles)
5. Contenedores de cosas
6. Cosas dentro de un contenedor
7. Organizaciones
8. Eventos

La lista anterior se puede enriquecer agregando categorías de concepto relacionadas con el dominio que se esta modelando. Otra técnica para encontrar conceptos idóneos es a partir de la identificación de frases nominales en las descripciones textuales del dominio del problema[10].

[10] Ver en el Capítulo 6 ejemplo para Aplicaciones Multimedia

Directrices para construir un modelo conceptual
1. Liste los conceptos idóneos (utilizando las técnicas antes mencionadas).
2. Dibújelos en un modelo conceptual.
3. Incorpore las asociaciones necesarias para registrar las relaciones entre los conceptos.
4. Agregue los atributos necesarios para cumplir con las necesidades de información.

3.2.3 Modelo de Análisis

El modelo conceptual nos ayuda a comprender los conceptos significativos en el dominio del problema desde un punto de vista orientado a objetos (Lenguaje del desarrollador). No obstante de acuerdo con la complejidad del proyecto que se asume en ocasiones el modelo conceptual no arroja suficiente luz sobre el dominio del problema, por lo que se hace necesario realizar otro modelo que nos servirá de punto de partida para entrar en el diseño, "El modelo de análisis".

A pesar de que el modelo del análisis hay un refinamiento de los requisitos, no se toman en cuenta el lenguaje de programación a usar en la construcción, la plataforma en la que se ejecutará la aplicación, los componentes prefabricados o reutilizables de otras aplicaciones, entre otras características que afectan al sistema, ya que el objetivo del análisis es comprender perfectamente los requisitos del software y no precisar cómo se implementará la solución.

La siguiente tabla compara brevemente el modelo de casos de uso, con el modelo de análisis:

Modelo de casos de uso	Modelo de análisis
Descrito con el lenguaje del cliente.	Descrito por el lenguaje del desarrollador.
Vista externa del sistema. Estructurado por los casos de usos: proporciona la estructura a la vista externa.	Vista interna del sistema. Estructurado por clases y paquetes estereotipados; proporciona la estructura a la vista interna.
Utilizado fundamentalmente como contrato entre el cliente y los desarrolladores sobre qué debería y que no debería hacer el sistema.	Utilizado fundamentalmente por los desarrolladores para comprender cómo debería darse forma al sistema, es decir, cómo debería ser diseñado e implementado.

Puede contener redundancias, inconsistencias, etc., entre requisitos.	No debería contener redundancias, inconsistencias, etc., entre requisitos.
Captura la funcionalidad del sistema incluida la funcionalidad significativa para la arquitectura.	Esboza cómo llevar a cabo la funcionalidad dentro del sistema incluida la funcionalidad significativa para la arquitectura; sirve como una primera aproximación del diseño.
Define casos de uso que se analizarán con más profundidad en el modelo de análisis.	Define realizaciones de casos de uso, y cada una de ellas representa el análisis de un caso de uso del modelo de casos de uso.

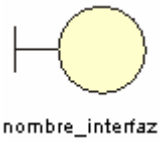
En el análisis podemos estructurar los requisitos de manera que nos facilite su comprensión, su preparación, su modificación y en general su mantenimiento. Esta estructura (basada en clases de análisis y paquetes) es independiente de la estructura que se dio a los requisitos (basada en casos de uso). Sin embargo existe una trazabilidad directa entre esas distintas estructuras, la cual se define entre casos de uso del modelo de casos de uso y realizaciones del caso de uso en el modelo de análisis

En este flujo se refinan y estructuran los requisitos obtenidos con anterioridad, profundizando el equipo del proyecto en el dominio de la aplicación lo que les permitirá una mayor comprensión del problema para modelar la solución.

Los artefactos que se construyen son:

- Modelo de análisis: Contiene clases del análisis y sus objetos organizados en paquetes que colaboran.
- Clases del análisis: Se centran en los requisitos funcionales y son evidentes en el dominio del problema porque representan conceptos y relaciones del dominio. Tienen atributos y entre ellas se establecen relaciones de asociación, agregación / composición, generalización / especialización y tipos asociativos. RUP propone clasificar a las clases en:

NOMBRE	CARATERÍSTICAS	REPRESENTACIÓN
Entidad	Modelan información que posee larga vida y que es a menudo persistente.	 nombre_entidad

Interfaz	Modelan la interacción entre el sistema y sus actores.	
Control	Coordinan la realización de uno o unos pocos casos de uso coordinando las actividades de los objetos que implementan la funcionalidad del caso de uso.	

En una aplicación cliente/servidor de tres capas, en la capa de usuario aparecen fundamentalmente clases interfaz ya que allí se ejecutan las aplicaciones del cliente. En la capa intermedia están las clases de control ya que en ella se agrupan los servicios que son compartidos por múltiples aplicaciones. En la capa servidor estarían las clases entidad porque allí se tiene la base de datos.

Los trabajadores, realizan las siguientes actividades:

- Análisis de la arquitectura: Identificar paquetes y clases del análisis.
- Analizar un caso de uso: Identificar clases cuyos objetos son necesarios para realizar el caso de uso y distribuir el comportamiento entre los objetos que participan en el caso de uso.
- Analizar una clase: Identificar atributos de las clases y relaciones entre ellas.
- Analizar un paquete: Garantizar alta cohesión y bajo acoplamiento de los paquetes y describir las relaciones entre ellos.

Artefactos a construir:

- Realización de casos de uso del análisis: Describe cómo se lleva a cabo y se ejecuta un caso de uso determinado en término de las clases del análisis y de sus objetos en interacción.
- Paquete de análisis: Organiza los artefactos del análisis en piezas manejables.
- Descripción de la arquitectura (vista del modelo de análisis): Muestra los artefactos significativos para la arquitectura (los anteriores).

3.2.4 Identificación de Clases

Las clases que se identifican están asociadas con el contexto del dominio del problema por lo que representan conceptos y relaciones.

Clases Interfaz

Al modelar la interacción entre el sistema y sus actores, pueden identificarse a partir de estos últimos:

- Al menos una clase que modele la interacción del actor usuario con el sistema, es decir, una clase para cada interacción actor-caso de uso sin preocuparse de que en la solución puede que se presente más de una pantalla dentro de un caso de uso para un mismo usuario.
- Una clase para cada sistema externo que será el responsable de la relación del sistema con cada uno de ellos.
- Una clase para cada actor que represente un dispositivo sobre el cual el sistema actúa o recibe información.

Estas dos últimas recomendaciones responden a un principio del paradigma de objetos en el cual se propone que en la definición de clases existe una alta cohesión interna de manera que si cambia algo solo se afecta(n) aquella(s) clase(s) que está(n) vinculada(s) con esa funcionalidad. En este caso, por ejemplo, si se modifica la estructura de la tabla que contiene todas las técnicas con las que trabaja la galería (tabla perteneciente a la base de datos del Sistema de Recursos Humanos) o, para el caso de un sistema de pronóstico del tiempo, cambia el dispositivo utilizar para medir la presión atmosférica, solo se afectarán las clases de interfaz porque estos cambios no afectan a la lógica de la aplicación.

En los ejemplos del capitulo 6 se muestran las clases que se obtendrían a partir del Diagrama de Casos de Uso.

Clases entidad

Estas clases modelan información que posee una larga vida y que a menudo es persistente y fenómenos, conceptos y sucesos que ocurren en el mundo real. La fuente principal de obtención son las clases entidades del negocio y el glosario de términos que se ha ido elaborando. Algunos autores proponen un estudio del texto, a partir de las frases nominales, de manera que los sustantivos representan objetos y clases.

Clases de control

Las clases de control coordinan el trabajo de uno o unos pocos casos de uso, coordinando las actividades de los objetos que implementan la funcionalidad del caso de uso, por lo que definen el flujo de control y las transacciones dentro de un caso de uso delegando el trabajo a otros objetos.

En principio se define una clase de control para cada caso de uso, aunque:

- Si el flujo de eventos (diferentes tareas a ser ejecutadas) de un CU, puede ser manejado por las clases de entidad e interfaz creadas, entonces no se justifica una clase de control, siendo la clase de interfaz la que coordina el CU.

- Si el flujo de eventos es complejo o el comportamiento dinámico que describe el CU, puede cambiar independientemente de la interfaz o de la información que almacena el sistema, entonces se debe definir una clase de control.
- Si el flujo de eventos o un subconjunto de un flujo de eventos puede re-usarse dentro de la realización de varios CU, entonces se debe definir una clase de control para ese flujo de eventos.
- Si un CU tiene más de un actor, puede crear una clase de control para cada actor si se asocian a funcionalidades independientes y se puede asegurar que el comportamiento de los actores cuando se relacionan con la clase de control, cambia.
- Si varios casos de uso tienen funcionalidades relacionadas (por ejemplo, por la herencia) o trabajan con los mismos objetos, podría definirse una clase de control para todos ellos.

El ciclo de vida de los objetos dentro de la realización de un caso de uso sería:

Objeto de interfaz:

Pueden aparecer en más de un CU si, por ejemplo, una ventana en particular aparece en más de un CU, sin embargo, por lo general se crean y finalizan dentro de la realización de un CU.

Objeto entidad:

Normalmente no son particulares de una realización de un CU.

Objeto de control:

Suelen encapsular el control asociado con un CU en concreto (por lo tanto, se crean cuando comienza la realización y se destruyen cuando termina), pero hay excepciones cuando:

- un objeto participa en más de una realización de CU diferentes.
- hay varios objetos de control que participan en una misma realización de un CU.
- una realización no requiere objetos de control.

3.2.5 Identificación de Atributos y Relaciones

Para la identificación de atributos y relaciones entre clases en el modelo de análisis, se siguen las mismas directrices que vimos en el modelo conceptual.

3.2.6 Diagrama de Clases del Análisis

Un Diagrama de Clases del Análisis es un artefacto en el que se representan los conceptos en un dominio del problema. Representa las cosas del mundo real, no de la implementación automatizada de estas cosas. En general se siguen directrices muy parecidas a las que usamos en la construcción del modelo conceptual.

Pasos para construir el Diagrama de Clases del Análisis

1. Crear una lista de conceptos significativos e interesantes del dominio.
2. Identificar las asociaciones.

Asociación: Relación entre dos conceptos que indican alguna conexión significativa e interesante entre ellos.

3. Identificar atributos.

Incluir solo los atributos que, según los requerimientos, son necesarios recordar para que se ejecute el caso de uso.

Recomendaciones:

- Definir como atributos las propiedades que tengan valores simples asociados.
- No usar atributos que sean llaves foráneas, si se necesita en la definición de una clase una llave foránea, esto indica que se requiere de una asociación entre las clases.
- No usar atributos complejos, si se necesita en la definición de una clase un objeto como atributo, esto indica que se requiere de una asociación entre las clases.

En el ejemplo del Capitulo 6 se muestra Diagrama de Clases del Análisis.

3.2.7 Organización en Paquetes

Los paquetes son un mecanismo de organización de elementos que subdividen el modelo en otros más pequeños que colaboran entre sí. Tienen que ser:

- Cohesivos: sus contenidos deben estar fuertemente relacionados.
- Débilmente acoplados: minimizar las dependencias entre paquetes.

Este particionamiento debe hacerse sobre la base de los requerimientos funcionales y el dominio del problema; y debe ser reconocible por las personas con conocimiento del dominio.

Para ello se propone asignar la mayor parte de un cierto número de casos de uso a un paquete concreto. Se pueden seguir los siguientes criterios de agrupamiento:

- Casos de uso requeridos para dar soporte a un determinado proceso de negocio.
- Casos de uso requeridos para dar soporte a un determinado actor del sistema.
- Casos de uso que están relacionados mediante relaciones de generalización y extensión.
- Casos de uso que se ejecutan en un nodo.

Hay un tipo de paquete que se conoce como paquete de servicio ya que contiene un conjunto de clases relacionadas funcionalmente y un conjunto coherente de acciones relacionadas funcionalmente, que se utilizan en varios casos de uso. Son altamente reutilizables, por lo que no se cumple con ellos el principio de bajo acoplamiento, y pueden emplearse en varias realizaciones de casos de uso diferentes.

En la figura 6 se presenta la notación de los paquetes y la relación de dependencia que se da entre ellos, aunque son válidas los mismos tipos de relaciones que se dan entre clases.

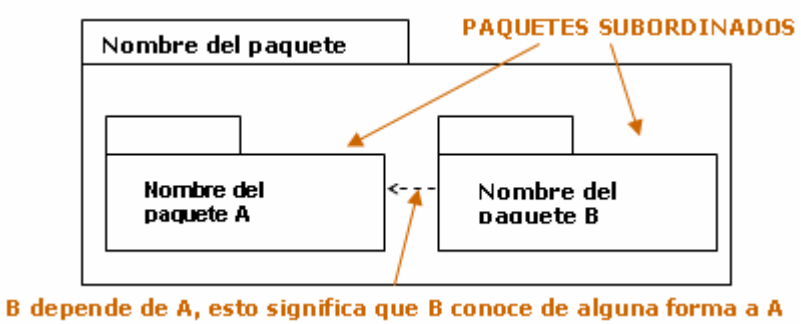


Figura 6 Notación de paquetes.

En la figura 7 se muestra como quedaría el agrupamiento para el caso de estudio de la galería de arte. Se pueden identificar paquetes de servicio (Correo), paquetes que agrupan casos de uso que dan soporte a un determinado actor del sistema (Evaluación de obras) y paquetes que agrupan a casos de uso relacionados con relaciones de extensión / inclusión (Registro de obras). En particular se describe el paquete de Registro de obras y se aprecia su relación con el paquete de Correo (se incluye el caso de uso que motiva la dependencia indicando su origen – from Correo).

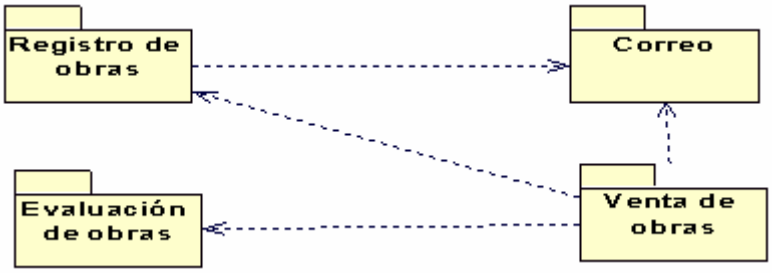


Figura 7 Agrupamiento en paquetes.

En el ejemplo del Capítulo 6 se muestra el contenido de los paquetes de acuerdo a los casos de uso que agrupa y las clases que se necesitan para realizarlos.

3.2.8 Diagrama de Colaboración

Un diagrama de colaboración destaca la organización de los objetos que participan en una interacción. Un diagrama de colaboración se construye colocando en primer lugar los objetos que participan en la colaboración como nodos del grafo. A continuación se representan los enlaces que conectan esos objetos como arcos del grafo. Por último, estos enlaces se adornan con los mensajes que envían y reciben los objetos.

Los diagramas de colaboración tienen dos características que los distinguen de los diagramas de secuencia. En primer lugar, el camino. Para indicar cómo se enlaza un objeto a otro, se puede

asociar un estereotipo de camino al extremo más lejano de un enlace (como «local», que indica que el objeto designado es local al emisor). Normalmente, sólo se necesita representar explícitamente el camino del enlace para los caminos local, parameter, global y self (pero no association).
En segundo lugar, está el número de secuencia. Para indicar la ordenación temporal de un mensaje, se precede de un número (comenzando con el mensaje número 1), que se incrementa secuencialmente por cada nuevo mensaje en el flujo de control (2, 3, etc.).

Para representar el anidamiento, se utiliza la numeración decimal; 1.1 es el primer mensaje dentro del mensaje 1; I 2 es el segundo mensaje dentro del mensaje 1; etc.). El anidamiento se puede representar a cualquier nivel de profundidad. Nótese también que, a través del mismo enlace, se pueden mostrar varios mensajes (posiblemente enviados desde distintas direcciones), y cada uno tendrá un número de secuencia único. La mayoría de las veces se modelarán flujos de control simple y secuencial. Sin embargo, también se pueden modelar flujos más complejos, que impliquen iteración y bifurcación. Una iteración representa una secuencia repetida de mensajes. Una iteración indica que el mensaje (y cualquier mensaje anidado) se repetirá de acuerdo con la expresión dada. Análogamente, una condición representa un mensaje cuya ejecución depende de la evaluación de una expresión booleana. Para modelar una condición, el número de secuencia de un mensaje se precede de una cláusula de condición, corno [x > 0 1 Los distintos caminos alternativos de una bifurcación tendrán el mismo número de secuencia, pero cada camino debe ser distinguible de forma única por una condición que no se solape con las otras. Tanto para la iteración como para la bifurcación, UML no impone el formato de la expresión entre corchetes; se puede utilizar pseudocódigo o la sintaxis de un Lenguaje de programación específico.

3.2.9 Diagrama de Secuencia

Un diagrama de secuencia destaca la ordenación temporal de los mensajes. un diagrama de secuencia se forma colocando en primer lugar los objetos que participan en la interacción en la parte superior del diagrama, a lo largo del eje X. Normalmente, se coloca a la izquierda el objeto que inicia la interacción, y los objetos subordinados a la derecha. A continuación, se colocan los mensajes que estos objetos envían y reciben a lo largo del eje Y, en orden de sucesión en el tiempo, desde arriba hasta abajo. Esto ofrece al lector una señal visual clara del flujo de control a lo largo del tiempo.
Los diagramas de secuencia[11] tienen dos características que los distinguen de los diagramas de colaboración. En primer lugar, está la línea de vida. La línea de vida de un objeto es la línea

[11] Ver en el Capítulo 6 ejemplos de Diagramas de Secuencia.

discontinua vertical que representa la existencia de un objeto a lo largo de un período de tiempo. La mayoría de los objetos que aparecen en un diagrama de interacción existirán mientras dure la interacción, así que los objetos se colocan en la parte superior del diagrama, con sus líneas de vida dibujadas desde arriba hasta abajo. Pueden crearse objetos durante la interacción. Sus líneas de vida comienzan con la recepción del mensaje estereotipado como create. Los objetos pueden destruirse durante la interacción. Sus líneas de vida acaban con la recepción del mensaje estereotipado como destroy (además se muestra la señal visual de una gran x que marca el final de sus vidas).

En segundo lugar, está el foco de control. El foco de control es un rectángulo delgado y estrecho que representa el período de tiempo durante el cual un objeto ejecuta una acción, bien sea directamente o a través de un procedimiento subordinado. La parte superior del rectángulo se alinea con el comienzo de la acción; la inferior se alinea con su terminación (y puede marcarse con un mensaje de retorno). También puede mostrarse el anidamiento de un foco de control (que puede estar causado por recursión, una Llamada a una operación propia, o una llamada desde otro objeto) colocando otro foco de control ligeramente a la derecha de su foco padre. Si se quiere ser especialmente preciso acerca de dónde se encuentra el foco de control, también se puede sombrear la región del rectángulo durante la cual el método del objeto está ejecutándose (y el control no ha pasado a otro objeto).

Mensajes

Envío de mensaje simple:

- Un solo hilo de ejecución.
- El objeto activo pasa el control al objeto pasivo.

Envío de mensaje síncrono:

- Sólo se desencadena la operación cuando el servidor acepta el mensaje.
- El cliente se bloquea hasta que el servidor lo acepta o rechaza.

Envío de mensaje asíncrono:

- Un mensaje asíncrono no interrumpe la ejecución del cliente
- El cliente envía el mensaje sin saber cuándo ni si se llevará a cabo.

Envío de Retorno:

- Se utiliza cuando, al mandar el mensaje, este devuelve algún valor de interés para un objeto.

3.2.10 Modelo de Diseño

Las actividades contempladas en el análisis comenzaron a adentrarnos en el problema a resolver por lo que a través de ellas representamos una vista interna del sistema en la que, usando el lenguaje de los desarrolladores se refinan los requisitos y se estructuran en base a clases y paquetes. Este proceso continúa en el diseño hasta obtener los objetos que interactúan para cumplir los requisitos funcionales y no funcionales obtenidos. Recordemos el propósito del análisis:

- Conseguir una comprensión más precisa de los requisitos, refinarlos y estructurarlos.
- Utilizar el lenguaje de los desarrolladores para analizar con profundidad los requisitos funcionales.
- Proporcionar una visión general del sistema.

$$\frac{\text{Abstracciones hechas en el modelo de análisis}}{\text{Riqueza de detalle del modelo de diseño}} = \frac{1}{\mathbf{5}}$$

El Modelo de Análisis puede considerarse como una primera aproximación al Modelo de Diseño. Es una entrada fundamental cuando se da forma al sistema en el Diseño y en la Implementación.
Durante el ciclo de desarrollo iterativo es posible pasar a la fase de diseño, una vez terminados estos documentos del análisis. Durante este paso se logra una solución lógica que se funda en el paradigma orientado a objetos. Su esencia es la elaboración de diagramas de interacción, que muestran gráficamente como los objetos se comunican entre ellos a fin de cumplir con los requerimientos.
El advenimiento de los diagramas de interacción nos permite dibujar diagramas de diseño de clases que resumen la definición de las clases (e interfaces) implementables en el software.

El diseño tiene el propósito de formular los modelos que se centran en los requisitos no funcionales y en el dominio de la solución y que prepara para la implementación y prueba del sistema. Pretende crear un plano del modelo de implementación, por lo que el grueso del esfuerzo está en las últimas iteraciones de elaboración y las primeras de construcción.

3.2.11 Papel del diseño en el ciclo de vida del software

RUP define el análisis y el diseño como un único flujo de trabajo en el que hay actividades que se realizan desde la fase de Inicio. Durante esta fase se analiza si es posible dar una solución que satisfaga a los requerimientos significativos de la arquitectura. En la fase de Elaboración se comienza con un análisis de los elementos significativos de la arquitectura como parte de la 1ra iteración de elaboración y en las siguientes iteraciones se refina la arquitectura hasta diseñar todos

sus elementos. En las restantes fases se hace algo de análisis y diseño vinculado con nuevos requerimientos que sean definidos y se decide implementar con los defectos que haya que corregir como consecuencia de las pruebas realizadas.

El diseño es el centro de atención al final de la fase de elaboración y el comienzo de las iteraciones de construcción. Esto contribuye a una arquitectura estable y sólida, y crear un plano del modelo de implementación. Durante la fase de construcción, cuando la arquitectura es estable y los requisitos están bien entendidos, el centro de atención se desplaza a la implementación.

El modelo de diseño está muy cercano al de implementación, lo que es natural para guardar y mantener el modelo de diseño a través del ciclo de vida completo del software.

En el diseño modelamos el sistema y encontramos su forma (incluida la arquitectura) para que soporte todos los requisitos, incluyendo los no funcionales y las restricciones que se le suponen. Una entrada esencial en el diseño es el resultado del análisis, o sea el modelo de análisis, que proporciona una comprensión detallada de los requisitos. Además impone una estructura del sistema que debemos esforzarnos por conservar lo más fielmente posible cuando demos forma al sistema.

Concretamente podemos definir como propósitos del diseño:

- Adquirir una comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución y concurrencia y tecnologías de interfaz de usuario.
- Crear una entrada apropiada y un punto de partida para actividades de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases.
- Descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo.
- Capturar las interfaces entre los subsistemas antes en el ciclo de vida del software, lo cual es muy útil cuando utilizamos interfaces como elementos de sincronización entre diferentes equipos de desarrollo.

A partir de los aspectos anteriores podemos establecer la siguiente comparación entre el modelo de análisis y el modelo de diseño:

Modelo de Análisis	Modelo de Diseño
Modelo conceptual (abstracción del sistema y permite aspectos de la implementación).	Modelo físico (plano de la implementación).

Genérico respecto al diseño (aplicable a varios diseños).	No genérico, específico para una implementación.
Tres estereotipos conceptuales sobre las clases: Control, Entidad e Interfaz.	Cualquier número de estereotipos (físicos) sobre las clases, dependiendo del lenguaje de implementación.
Dinámico (no muy centrado en la secuencia).	Dinámico (centrado en las secuencias).
Bosquejo del diseño del sistema, incluyendo su arquitectura.	Manifiesto del diseño del sistema, incluyendo su arquitectura (una de sus vistas).
Puede no estar mantenido durante todo el ciclo de vida del software.	Debe ser mantenido durante todo el ciclo de vida del software.
Define una estructura que es una entrada esencial para modelar el sistema, incluyendo la creación del modelo de diseño.	Da forma al sistema mientras que intenta preservarla estructura definida por el modelo de análisis lo más posible.

Artefactos resultantes del Modelo de Diseño

Los artefactos son el resultado de un trabajador al jugar su rol dentro del flujo de trabajo. En la siguiente figura se muestra los artefactos que se obtienen de este flujo de trabajo de acuerdo al trabajador que los obtiene.

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en como los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar.

Sirve de abstracción de la implementación y es utilizada como entrada fundamental de las actividades de implementación.

Se representa por un sistema de diseño que denota el subsistema de nivel más alto del modelo. La utilización de otro subsistema es, entonces, una forma de organización del modelo de diseño en porciones más manejables.

En el modelo del diseño, los casos de uso son realizados por las clases del diseño y sus objetos. Esto se representa por colaboraciones en el modelo de diseño y denota la realización de casos de uso del diseño. La realización de casos de uso del diseño es diferente de la realización de casos de uso de análisis.

Los artefactos del Modelo de Diseño son: Modelo de Despliegue, Descripción de la Arquitectura, Realización de Casos de Uso, Clase del Diseño, Subsistema de Diseño, Interfaz.

3.2.12 Mapa de Navegación

Es el tratamiento comunicacional de los contenidos, en otras palabras es la organización de la presentación de la información expresada en un diagrama.

La importancia de elaborar un mapa de navegación del sitio web radica en la comprensión del orden de presentación de las pantallas con los contenidos (páginas web) y la flexibilidad de moverse entre ellas (hipervínculos).

En el diseño del mapa de navegación se debe:

- Seleccionar la pantalla de entrada al sitio web presentación (página web: index.html).
- Ordenar de manera jerarquizada las pantallas con los contenidos (por niveles o categorías).
- Establecer los vínculos entre pantallas (páginas web) permitiendo una navegación hipertextual.

En esta estructura de mapa de navegación (Sitio Web) tenemos dos niveles:

El primero, la navegación es posible entre las actividades e index de manera hipertextual. El segundo, la navegación hipertextual es posible entre actividad y sus páginas correspondientes (contenido teórico, laboratorio, aplicación, evaluación, glosario y enlaces); por otro lado, permite ir desde cualquiera de estas páginas hasta las otras actividades entrando en el primer nivel de navegación.

El modelo de navegación de una aplicación Web comprende la especificación de qué objetos pueden ser visitados mediante la navegación a través de la aplicación Web y las asociaciones entre ellos. Los modelos de la navegación son representados por los diagramas de clases estereotipadas.

3.2.13 Prototipo físico de interfaz de usuario

El Diagrama de Presentación es el que debe mostrar las formas de organización visual de los contenidos en las páginas principales, por ejemplo: la página inicial, las páginas interiores, páginas de productos, etc. Este diagrama no pretende representar el diseño gráfico o diseño visual en detalle, sino especificar el esqueleto organizativo de la interfaz.

3.2.14 Modelo de Despliegue

El Modelo de Despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de computo. El Modelo de

Despliegue se utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una influencia principal en su diseño.

Podemos observar lo siguiente sobre Modelo de Despliegue[12]:

- Cada nodo representa un recurso de cómputo, normalmente un procesador o un dispositivo hardware similar.
- Los nodos poseen relaciones que representan medios de comunicación entre ellos, tales como Internet, Intranet, bus, y similares.
- El modelo de despliegue puede describir diferentes configuraciones de red, incluidas las configuraciones para pruebas y para simulación.
- La funcionalidad (los procesos) de un nodo se define por los componentes que se distribuyen sobre ese nodo.
- El modelo de despliegue en si mismo representa una correspondencia entre la arquitectura software y la arquitectura del sistema (el hardware).

3.2.15 Vistas arquitectónicas

La arquitectura es el esqueleto o base de una aplicación, en esta se analiza la aplicación desde varios puntos de vista. En la arquitectura aparecen los artefactos mas importantes y diferentes, para establecer un esquema de cómo deben ser los próximos artefactos a construir, o sea es puramente un semi-molde al que se deben ajustar sin muchos cambios los restantes artefactos a construir en la aplicación o solución. De obtenerse un artefacto demasiado diferente a los demás, este formaría parte de la arquitectura. En RUP, esta se compone de 4+1 vistas fundamentalmente, se dice que 4 de estas vistas están regidas por 1 vista rectora, estas vistas son: Vista de Casos de Uso, Vista Lógica, Vista de Procesos, Vista de Implementación, Vista de Despliegue. Estas vistas son la parte esencial de los que se obtuvo en las etapas que mas tributan a la arquitectura.

3.2.16 Arquitectura del sistema

Se representa a través de 4 + 1 vista, es decir 4 de estas vistas están regidas por una rectora. La rectora se considera la vista de casos de uso, y las restantes son la vista lógica, la vista de procesos, la vista de despliegue y la vista de implementación.

Vista de Casos de Uso: Esta vista representa un subconjunto del artefacto Modelo de casos de uso y lista los casos de usos o escenarios del modelo de casos de uso más significativos, con las funcionalidades centrales del sistema.

[12] Ver ejemplo en el Capítulo 6

Vista Lógica: Esta vista representa un subconjunto del artefacto Modelo de Diseño, la cual representa los elementos de diseño más importantes para la arquitectura del sistema. Describe las realizaciones de los casos de uso más relevantes y de las clases más importantes; su organización en paquetes y subsistemas.

Vista de Procesos: Esta vista suministra una base para la comprensión de la organización de los procesos de un sistema, ilustrados en el mapeo de las clases y subsistemas en procesos e hilos. Solo suele usarse cuando el sistema presenta procesos concurrentes o hilos.

Vista de Despliegue: Esta vista suministra una base para la comprensión de la distribución física de un sistema a través de nodos. Suele utilizarse cuando el sistema está distribuido.

Vista de Implementación: Esta vista describe la descomposición del software en capas y subsistemas de implementación. También provee una vista de la trazabilidad de los elementos de diseño de la vista lógica ahora para la implementación.

3.2.17 Estilos arquitectónicos

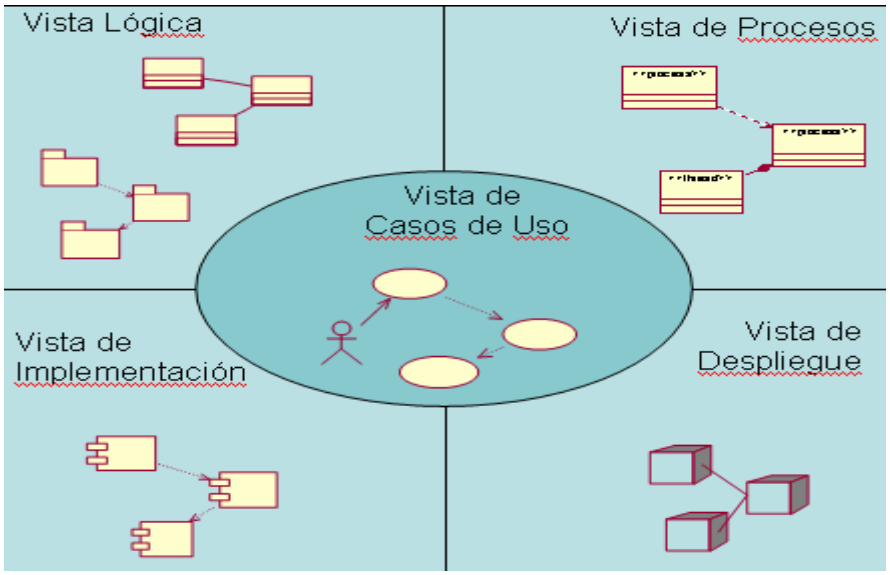
Arquitecturas en Capas: Se define cómo organizar el modelo de diseño a través de capas, que pueden estar físicamente distribuidas, lo cual quiere decir que los componentes de una capa sólo pueden hacer referencia a componentes en capas inmediatamente inferiores.

Arquitecturas Orientadas a Objetos: Los componentes de este estilo se basan en principios OO: encapsulamiento, herencia y polimorfismo. En general la distribución de objetos es transparente, o sea que apenas importa si los objetos son locales o remotos.

Arquitecturas Orientadas a Servicios: Es un patrón en el que los recursos que están interconectados en una red se conciben como servicios accesibles por terceros a través de una interfaz estándar. Este modelo posee bajo grado de acoplamiento entre componentes junto a una mayor flexibilidad ante cambios futuros ya que un cambio en el diseño interno puede ser factible sin necesidad de modificar el servicio.

Vistas Arquitectónicas	Etapas a la que Tributan
Vista de Casos de Uso	Requisitos
Vista Lógica	Análisis y Diseño
Vista de Procesos	Análisis y Diseño
Vista de Implementación	Implementación
Vista de Despliegue	Análisis y Diseño

Como se corrobora, la etapa Análisis y Diseño es la que mas aporta a la arquitectura puesto existen tres vistas arquitectónicas para la misma (ver Figura). Para la obtención de los artefactos que llenan estas vistas se realiza un trabajo arduo, ya que son el resultado de un análisis profundo del arquitecto, pero se cuenta con herramientas poderosas que son fruto de la experiencia y la heurística acumulada a través del tiempo por los estudiosos de la IGS y la POO, estas herramientas son el conjunto de los Patrones Arquitectónicos, los cuales tienen aplicación en todas las vistas incluso en varias de ellas al mismo tiempo, estos influyen agilizando y estandarizando el trabajo, ya que muestran formas de solución a muchos de los problemas particulares que hay que enfrentar a la hora de realizar la arquitectura de una aplicación. Además los patrones muestran el como de la transición de el diseño a la ejecución del diseño. Actualmente siguen surgiendo nuevos patrones incluso específicos para una plataforma.



Arquitectura clásica de tres capas

Una arquitectura común de los sistemas de información que abarca una interfaz para el usuario y el almacenamiento persistente de datos se conoce con el nombre de arquitectura de tres capas. He aquí una descripción clásica de las capas verticales:

1. Presentación: ventanas, reportes, etcétera.
2. Lógica de aplicaciones: tareas y reglas que rigen el proceso.
3. Almacenamiento: mecanismo de almacenamiento persistente.

La calidad tan especial de la arquitectura de tres capas consiste en aislar la lógica de la aplicación y en convertirla en una capa intermedia bien definida y lógica del software. En la capa de presentación se realiza relativamente poco procesamiento de la aplicación; las ventanas envían a la

capa intermedia peticiones de trabajo. Y este se comunica con la capa de almacenamiento del extremo posterior.

3.2.18 Patrones

Los patrones de diseño han contribuido a dar flexibilidad y extensibilidad a nuestros diseños. Pero en adición, han demostrado ser una forma muy útil (exitosa) de reutilizar diseño, ya que ellos no sólo nombran, abstraen e identifican aspectos claves de estructuras comunes de diseño, sino que generalmente son descritos en una forma específica documental, haciendo su comprensión y aplicación fácil para el conjunto de desarrolladores.

Patrones básicos de GRASP

Patrón Experto

Asignar una responsabilidad al experto en la información.

Patrón Creador

¿Quién debería ser responsable de crear una nueva instancia de alguna clase?

Patrón Controlador

¿Quién debería encargarse de atender un evento del sistema?

Asignar la responsabilidad del manejo de mensajes de los eventos del sistema a una clase.

Patrón Bajo Acoplamiento

¿Cómo dar soporte a una mínima dependencia y a un aumento de la reutilización?

Una clase con bajo acoplamiento no depende de “muchas otras” clases. Las clases con alto acoplamiento recurren a muchas clases y no es conveniente porque son más difíciles de mantener, entender y reutilizar.

Patrones de diseño

Patrones de Creación

- Abstract Factory
- Builder
- Factory Method
- Prototype
- Singleton

Patrones Estructurales

- Adapter
- Bridge

- Composite
- Decorator
- Facade
- Flyweight
- Proxy

Patrones de comportamiento

- Chain of Responsibility.
- Command
- Interpreter
- Iterator
- MediatorMemento
- Observer
- State
- Strategy
- Template Method
- Visitor

Beneficios de los Patrones:

- Contribuyen a reutilizar diseño, identificando aspectos claves de la estructura de un diseño que puede ser aplicado en una gran cantidad de situaciones. La importancia de la reutilización del diseño no es despreciable, ya que ésta nos provee de numerosas ventajas: reduce los esfuerzos de desarrollo y mantenimiento, mejora la seguridad, eficiencia y consistencia de nuestros diseños, y nos proporciona un considerable ahorro en la inversión.
- Mejoran (aumentan, elevan) la flexibilidad, modularidad y extensibilidad, factores internos e íntimamente relacionados con la calidad percibida por el usuario.
- Incrementan el vocabulario de diseño, ayudando a diseñar desde un mayor nivel de abstracción.

Niveles de Abstracción

Si bien la mayor parte de los patrones tiene su aplicación directa en código, no significa que sea el único nivel al que se manejan. Comentemos un ejemplo del sitio de Microsoft, que aplica un patrón que es muy conocido por los desarrolladores, especialmente en los últimos años.

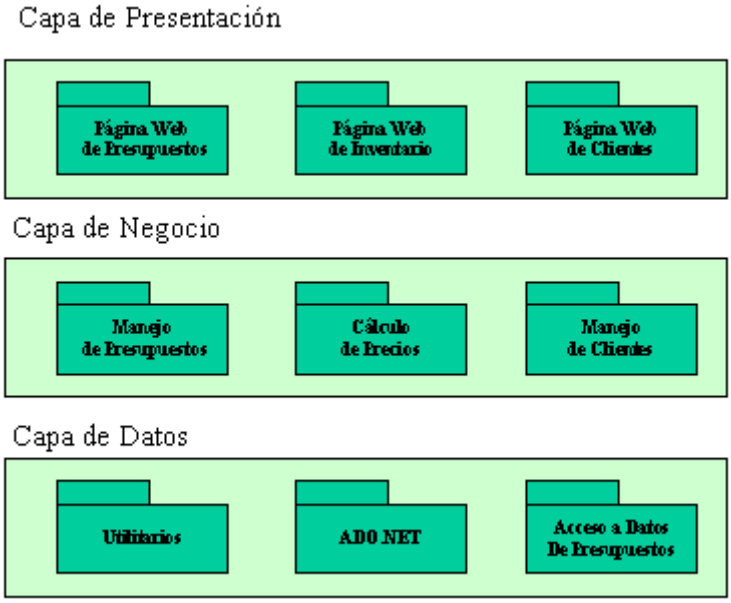
Estamos desarrollando un sistema empresarial, donde queremos manejar la información de presupuestos para los clientes de la empresa. Necesitamos conocer los precios internos, armar el presupuesto, tener en

cuenta el inventario de los productos, y contemplar los datos e historia del cliente que pide el presupuesto. La información debe estar disponible para ser consultada por la web, y provenir de distintas fuentes de datos.

Querríamos organizar la arquitectura del sistema para que sea flexible, por ejemplo, que contemple las distintas formas de acceder a los datos, y las formas de procesarlos, y por último, los distintos modos de visualizar el resultado.

Así planteado, es una especificación muy general, pero que nos sirve para presentar un patrón conocido de arquitectura: el patrón de capas.

Podemos tomar la decisión de separar la aplicación según este esquema:



Esta solución termina aplicando el patrón Layers[13](capas), que tanta veces hemos encontrado (hasta se utiliza en la arquitectura de redes, donde el modelo ISO se basa justamente en dividir el trabajo en las capas de transporte, aplicación y otras).

Formalizando la definición del patrón, podemos encontrar:

Contexto

Estamos trabajando en un sistema grande y complejo. Y queremos manejar la complejidad y la descomposición.

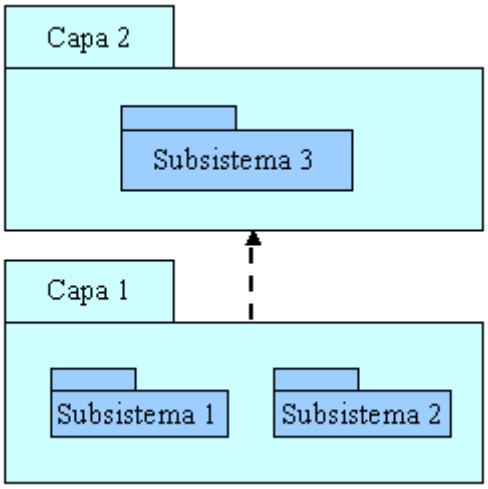
Problema

Cómo estructurar la información para soportar los requerimientos de mantenimiento, reusabilidad, escalabilidad, y robustez.

[13] Ver anexo 9

Solución

Componer la solución en una serie de capas. Cada capa debe ocuparse de un nivel del problema, y debe tener poca cohesión con las demás.



Este patrón tiene consecuencias no evidentes: el cambio en una capa, debería alterar en poco los cambios en las otras capas. La prueba ácida, en nuestro ejemplo original, podría ser: al cambiar las fuentes de datos (las bases de datos usadas), poco debería afectar a la capa de presentación. Y si el día de mañana queremos incorporar una capa de presentación distinta (como un sistema de atención automática por teléfono, usando un IVR), no debería alterar a las otras capas ya desarrolladas. En definitiva, como en el ejemplo del modelo ISO de redes, el cambiar la implementación de una capa, debe tener los mínimos efectos en el resto de la aplicación.

3.2.19 Descripción de la Arquitectura (vista del Modelo de Diseño)

La descripción de la arquitectura contiene una vista de la arquitectura del modelo de diseño que muestra sus artefactos relevantes para la arquitectura. Suelen considerarse significativos para la arquitectura los siguientes artefactos del modelo de diseño:

1. La descomposición del modelo de diseño en subsistemas, sus interfaces, y las dependencias entre ellos. Esta descomposición es muy significativa para la arquitectura en general, debido a que los subsistemas y sus interfaces constituyen la estructura fundamental del sistema.
2. Clases de diseño fundamentales como clases que poseen una traza con clases del análisis significativas, clases activas, y clases del diseño que sean generales y centrales.
3. Realizaciones de casos de uso- diseño que describen alguna funcionalidad importante y crítica que debe desarrollarse pronto dentro del ciclo de vida.

3.2.20 Descripción de la Arquitectura (vista del Modelo de Despliegue)

La descripción de la arquitectura contiene una vista de la arquitectura del modelo de despliegue, que muestra sus artefactos relevantes para la arquitectura. Debido a su importancia, deberían mostrarse todos los aspectos del modelo de despliegue en la vista arquitectónica, incluyendo la correspondencia de los componentes sobre los nodos tal como se identificó durante la implementación.

3.2.21 Realización de Casos de Uso del Diseño

Es una colaboración en el modelo de diseño que describe como se realiza un caso de uso especifico, y como se ejecuta en términos de casos de uso del diseño. Una realización de caso de uso del diseño proporciona una traza directa a una realización de caso de uso del análisis en el modelo de análisis.

Cuando el modelo de análisis no va a mantenerse a lo largo del ciclo de vida del software pero en cambio se utiliza solo para crear un buen diseño, no tendremos realización de caso de uso-análisis. La dependencia de traza de una realización de caso de uso-diseño irá en este caso directamente hasta el caso de uso en el modelo de casos de uso.

Una realización de caso de uso del diseño tiene una descripción de flujos de eventos textual, diagramas de clases que muestra sus clases de diseño participantes y diagramas de interacción que muestran la realización de un flujo o escenarios concretos de un caso de uso en términos de interacción entre objetos del diseño.

Proporciona una realización física de la realización de caso de uso de análisis y también gestiona muchos requisitos no funcionales capturados en la realización de casos de uso del análisis. Por consiguiente una realización de casos de uso de diseño puede posponer el manejo de algunos requisitos hasta las subsiguientes actividades de implementación anotándolas como requisitos de implementación en la realización.

La realización de casos de uso del análisis se efectúa a través de una colaboración como dijimos anteriormente en el cual intervienen el diagrama de clases y los diagramas de interacción.

3.2.22 Diagrama de Clases

Los diagramas de clases son los más utilizados en el modelado de sistemas orientados a objetos. Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Gráficamente, un diagrama de clases es una colección de nodos y arcos.

Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema Principalmente, esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas. Los diagramas de clases también son la base para un par de diagramas relacionados: los diagramas de componentes y los diagramas de despliegue. Los diagramas de clases son importantes no sólo para visualizar, especificar y documentar modelos estructurales, sino también para construir sistemas ejecutables, aplicando ingeniería directa e inversa.

Contienen normalmente los siguientes elementos:

1. Clases.
2. Interfaces.
3. Colaboraciones.
4. Relaciones de dependencia, generalización y asociación.

Al igual que los demás diagramas, los diagramas de clases pueden contener notas y restricciones. Los diagramas de clases también pueden contener paquetes o subsistemas, los cuales se usan para agrupar los elementos de un modelo en partes más grandes. A veces se colocarán instancias en los diagramas de clases, especialmente cuando se quiera mostrar el tipo (posiblemente dinámico) de una instancia. Nota: Los diagramas de componentes y los diagramas de despliegue son similares a los diagramas de clases, excepto que en lugar de clases contienen componentes y nodos, respectivamente.

Estrategia para construir un Diagrama de Clases del Diseño

Para preparar un diagrama de clases orientado al diseño:

- Identifique todas las clases que participan en la solución del software. Para ello analice los diagramas de interacción.
- Dibújelas en un diagrama de clases.
- Duplique los atributos provenientes de los conceptos asociados del modelo conceptual.
- Agregue los nombres de los métodos analizando los diagramas de interacción.
- Incorpore la información sobre los tipos a los atributos y a los métodos.
- Agregue las asociaciones necesarias para dar soporte a la visibilidad requerida de los atributos.
- Agregue flechas de navegabilidad a las asociaciones para indicar la dirección de la visibilidad de los atributos.
- Agregue las líneas de relaciones de dependencia para indicar la visibilidad no relacionada con los atributos.

3.2.23 Diagramas de Interacción

Los diagramas de secuencia y los diagramas de colaboración (ambos llamados diagramas de interacción) son dos de los cinco tipos de diagramas de UML que se utilizan para modelar los aspectos dinámicos de los sistemas. Un diagrama de interacción muestra una interacción, que consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Un diagrama de secuencia es un diagrama de interacción que destaca la ordenación temporal de los mensajes; un diagrama de colaboración es un diagrama de interacción que destaca la organización estructural de los objetos que envían y reciben mensajes.

Los diagramas de interacción se utilizan para modelar los aspectos dinámicos de un sistema. La mayoría de las veces, esto implica modelar instancias concretas o prototípicas de clases, interfaces, componentes y nodos, junto con los mensajes enviados entre ellos, todo en el contexto de un escenario que ilustra un comportamiento. Los diagramas de interacción pueden utilizarse para visualizar, especificar, construir y documentar la dinámica de una sociedad particular de objetos, o se pueden utilizar para modelar un flujo de control particular de un caso de uso.

Los diagramas de interacción no son sólo importantes para modelar los aspectos dinámicos de un sistema, sino también para construir sistemas ejecutables por medio de ingeniería directa e inversa.

Un Diagrama de Interacción muestra una interacción, que consta de un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Un diagrama de secuencia es un diagrama de interacción que destaca la ordenación temporal de los mensajes. Gráficamente, un diagrama de secuencia es una tabla que representa objetos, dispuestos a lo largo del eje X, y mensajes, ordenados según se suceden en el tiempo, a lo largo del eje Y. Un diagrama de colaboración es un diagrama de interacción que destaca la organización estructural de los objetos que envían y reciben mensajes. Gráficamente, un diagrama de colaboración es una colección de nodos y arcos.

Los diagramas de interacción contienen:

1. Objetos
2. Enlaces
3. Mensajes

Un diagrama de interacción es básicamente una proyección de los elementos de una interacción. La semántica del contexto de una interacción, los objetos y roles, enlaces, mensajes y secuenciación

se aplican a los diagramas de interacción. Al igual que los demás diagramas, los diagramas de interacción pueden contener notas y restricciones.

3.2.24 Diagramas de Colaboración

Para preparar un diagrama de colaboración:

- Elabore un diagrama por cada operación del sistema durante el ciclo actual de Desarrollo.
- En cada rnensaje del sistema, dibuje un diagrama incluyéndolo como mensaje inicial.
- Si el diagrama se torna complejo (por ejemplo, si no cabe holgadamente en una hoja de papel de 8.5 x 1 I), divídalo en diagramas más pequeños.
- Diseñe un sistema de objetos interactivos que realicen las tareas, usando corno punto de partida las responsabilidades del contrato de operación las poscondiciones y la descripción de casos de uso. Aplique el GRASP y otros patrones para desarrollar un buen diseño.

Para modelar una colaboración:

1. Hay que identificar los mecanismos que se quieren modelar. Un mecanismo representa una función o comportamiento de la parte del sistema que se está modelando que resulta de la interacción de una sociedad de clases, interfaces y otros elementos.
2. Para cada mecanismo, hay que identificar las clases, interfaces y otras colaboraciones que participan en esta colaboración. Asimismo, hay que identificar las relaciones entre estos elementos. Hay que usar escenarios para recorrer la interacción entre estos elementos. Durante el recorrido, se descubrirán partes del modelo que faltaban y partes que eran semánticamente incorrectas.
3. Hay que asegurarse de rellenar estos elementos con su contenido. Para las clases, hay que comenzar obteniendo un reparto equilibrado de responsabilidades. Después, a lo largo del tiempo, hay que convertir éstas en atributos y operaciones concretos.

3.2.25 Diagramas de Secuencia

Para elaborar diagramas de secuencia de un sistema que describan el curso normal de los eventos en un caso de uso:

1. Trace una línea que represente el sistema corno una caja negra.
2. Identifique los actores que operan directamente sobre el sistema. Trace una línea para cada uno de ellos.
3. A partir del curso normal de los eventos del caso de uso identifique los eventos ("externos") del sistema que son generados por los actores. Muéstrelos gráficamente en el diagrama.

4. A la izquierda del diagrama puede incluir o no el texto del caso de uso.

3.2.26 Requisitos de Implementación

Los requisitos de implementación son una descripción textual que recoge requisitos, tales como los requisitos no funcionales, sobre una realización de casos de uso. Nos referimos a requisitos que se capturan solo en la fase de diseño, pero que es mejor tratar en la implementación. Algunos de estos requisitos pueden haber sido identificados en flujos de trabajos anteriores y por tanto solo se cambian a una realización de caso de uso.

3.2.27 Flujo de Sucesos del Diseño

Es una descripción textual que explica y complementa los diagramas y a sus etiquetas.

En sentido general la realización de casos de uso esta basada en una parte estática que se describe a partir del diagrama de clases del caso de uso y una parte dinámica que se describe a través de los diagramas de interacción que pueden ser diagramas de secuencia y diagrama de colaboración.

3.2.28 Clase del Diseño

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

- El lenguaje utilizado para especificar una clase del diseño es lo mismo que el lenguaje de programación.
- Las relaciones de aquellas clases del diseño implicadas por otras clases, a menudo tienen un significado directo cuando la clase es implementada.
- Los métodos de una clase del diseño tienen correspondencia directa con el correspondiente método en la implementación de las clases.
- Una clase de diseño puede posponer el manejo de algunos requisitos para las siguientes actividades de implementación, indicándolos como requisitos de implementación de la clase.
- Una clase de diseño puede proporcionar interfaces si tiene sentido hacerlo en el lenguaje de programación.

3.2.29 Subsistema del Diseño

Son una forma de organizar los artefactos del modelo de diseño en piezas más manejables. Puede constar de clases del diseño, realizaciones de casos de uso, interfaces y otros subsistemas. Pueden proporcionar interfaces que representan la funcionalidad que exportan en términos de operaciones.

Los subsistemas pueden representar una separación de aspectos del diseño. También representan componentes de grano grueso en la implementación del sistema, es decir componentes que proporcionan varios interfaces compuestos a partir de otros varios componentes de grano más fino, como los que especifican clases de implementación individuales.

Los subsistemas pueden representar productos software reutilizado que han sido encapsulados en ellos. Además de representar sistemas heredados.

3.2.30 Interfaz

Las interfaces se utilizan para especificar las operaciones que proporcionan las clases y los subsistemas del diseño.

Una clase de diseño que proporcione una interfaz debe proporcionar también métodos que realicen las operaciones de la interfaz. Un subsistema que proporcione una interfaz debe contener también clases del diseño u otros subsistemas (recursivamente) que proporcionen la interfaz.

3.2.31 Técnicas comunes de modelado

Cuando se crea un diagrama de clases, se está modelando una parte de los elementos y relaciones que configuran la vista de diseño del sistema Por esta razón, cada diagrama de clases debe centrarse en una colaboración cada vez.

El modelado es importante, pero hay que recordar que el producto principal de un equipo de desarrollo es software, no son los diagramas. Por supuesto, la razón por la que se crean modelos es para entregar de forma predecible, en el momento oportuno, el software adecuado que satisfaga los objetivos cambiantes de los usuarios y la empresa. Por esta razón, es importante que los modelos que se creen y las implementaciones que se desplieguen se correspondan entre sí, de forma que se minimice o incluso se elimine el coste de mantener sincronizados los modelos y las implementaciones.

Para algunos usos de UML, los modelos que se realicen nunca se corresponderán con un código. Por ejemplo, si se modela un proceso de negocio con diagramas de actividades, muchas de las actividades modeladas involucrarán a gente, no a computadores. En otros casos, se modelarán sistemas cuyas partes sean, desde un nivel dado de abstracción, una pieza de hardware (aunque a otro nivel de abstracción, seguro que este hardware puede contener un computador y software empotrado).

3.2.32 Diseño de la Base de Datos (BD)

Las Bases de Datos necesitan de una definición de su estructura que le permita almacenar datos, reconocer el contenido, y recuperar la información. La estructura tiene que ser desarrollada para la necesidad de las aplicaciones que la usaran, esto nos puede ayudar a realizar un proceso del negocio para alcanzar un valor agregado para el cliente.

La puesta en práctica de la base de datos es el paso final en el desarrollo de aplicaciones de soporte del negocio. Tiene que conformarse con los requisitos del proceso del negocio, que es la primera abstracción de la vista de la base de datos.

Pasos que guían el Diseño de la Base de datos a partir de un modelo orientado a objetos.

Los pasos en el diseño de la BD son:

1. Definir las clases persistentes.
2. Refinar las clases.
3. Clasificar las clases y los atributos.
4. Realizar el diagrama de clases.
5. Realizar el diagrama de transición de estado.
6. Obtener las restricciones estáticas y las fórmulas dinámicas.
7. Convertir las clases al medio de almacenamiento.

En el paso relacionado con la conversión de las clases al medio de almacenamiento es que se tiene en cuenta hacía qué lugar se guardarán los objetos, por lo que se particulariza qué hacer en cada caso.

3.3 Flujo de Trabajo de Implementación

En la implementación se empieza con el resultado del diseño y se implementa el sistema en términos de componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares.

El flujo de trabajo de diseño se propone crear un plano del modelo de implementación, por lo que sus últimas actividades están vinculadas a la creación del modelo de despliegue. El flujo de trabajo de implementación describe cómo los elementos del modelo del diseño se implementan en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue.

Los diagramas de despliegue y componentes conforman lo que se conoce como un modelo de implementación al describir los componentes a construir y su organización y dependencia entre nodos físicos en los que funcionará a aplicación.

El flujo de implementación esta fuertemente determinado por el lenguaje de programación. En la fase de elaboración va encaminado a implementar la arquitectura que se ha definido.

Este flujo tiene los siguientes objetivos:

- Definir la organización del sistema en términos de Subsistemas de Implementación organizados en capas.
- Implementar los elementos de diseño en términos de "Elementos de Implementación" (ficheros Fuentes, binarios, ejecutables y otros).
- Probar los componentes desarrollados independientemente como unidades.
- Integrar los resultados producidos por desarrolladores independientes o equipos en un sistema ejecutable.

3.3.1 Subsistema de Implementación

Una colección de componentes y otros subsistemas de implementación usados para estructurar el modelo de implementación y dividirlos en pequeñas partes que pueden ser integradas y probadas de forma separada.

3.3.2 Componente

Parte modular de un sistema, desplegable y reemplazable que encapsula implementación y un conjunto de interfaces y proporciona la realización de los mismos. Un componente[14] típicamente contiene clases y puede ser implementado por uno o más artefactos (ficheros ejecutables, binarios, etc.).

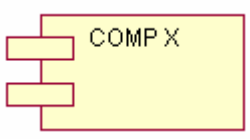
Algunos estereotipos estándar de componentes son los siguientes:

Ejecutable: Es un programa que se puede ejecutar en un nodo.

- Biblioteca: Es una biblioteca de objetos estática o dinámica.
- Tabla: Es una tabla de una BD.
- Archivo: Es un fichero que contiene código fuente o datos.
- Documento: Es un documento.
- Página Web: Es una página que se obtiene de la ejecución del sistema.

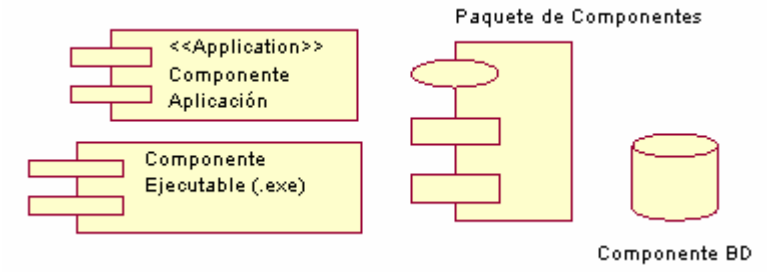
Se representa:

[14] Ver características de los componentes en anexo 9

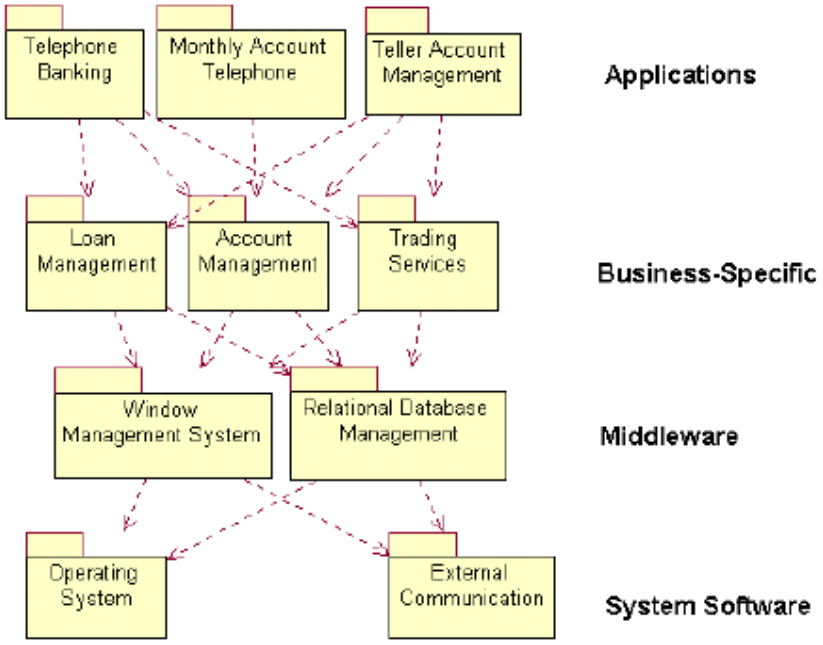


Puede llevar asociado una Interfaz, para más detalle.

Existen varias representaciones en dependencia del estereotipo que corresponde al tipo de componente que sea:



En la figura se muestra como se puede estructurar el modelo de implementación por capas. La capa de aplicación agrupa servicios específicos de la aplicación. La capa de negocio especifica componentes que se usan para el negocio y que se pueden usar en varias aplicaciones. La capa media contiene componentes de administración de base de datos, componentes OLE. Por ultimo la capa de software contiene componentes específicos de hardware y sistema operativo.



3.3.3 Mapeo del Diseño al Código

Cada clase en diseño es implementada a través de su codificación en un lenguaje de programación y/o mediante el uso de componentes pre-existentes. La correspondencia exacta entre una clase de diseño y su implementación correspondiente dependerá del lenguaje de programación. Por ejemplo,

en lenguajes orientados a objetos, como pudiera ser el caso de C++ o Java, una clase de diseño puede corresponder a una o varias clases. En Visual Basic, una clase de diseño corresponde a un componente de un tipo específico; por ejemplo, un módulo de clase, un formulario, o un control.
El modelo de diseño podrá ser más o menos cercano al modelo de implementación dependiendo de cómo se mapeen sus clases a clases o construcciones similares en el lenguaje de implementación. Por ejemplo, se le puede dar a las clases de implementación una relación 1:1 respecto a las clases de diseño. Ello conducirá a un modelo de diseño con rastreabilidad transparente al modelo de implementación, el cual es apropiado en ambientes de "ingeniería de viaje redondo" (ingeniería e ingeniería inversa).

Otra alternativa es permitir que una clase de diseño sea una abstracción de diversas clases de implementación. Cuando se usa esta perspectiva, es recomendable mapear cada clase de diseño con una clase "principal" (*head class*) que, a su vez, haga uso de distintas clases "auxiliares" (*help classes*) para realizar sus servicios. Se pueden emplear clases auxiliares para implementar atributos complejos o para construir estructuras de datos que se requieran para implementar una operación. En el diseño no se necesitará modelar las clases auxiliares, sino modelar algunos de los atributos, relaciones y operaciones definidas por la clase principal, en la medida en que el modelo cumpla con su propósito y sea una buena abstracción del código fuente. El cómo se relacionarán las clases de diseño con las de implementación se deberá decidir antes de que comience el diseño y deberá ser documento en las *Guías de Diseño*.

3.4 Flujo de Trabajo de Prueba

Las pruebas es una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente. La prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación.
RUP propone que en cada una de las fases las pruebas se comporten de la siguiente forma:

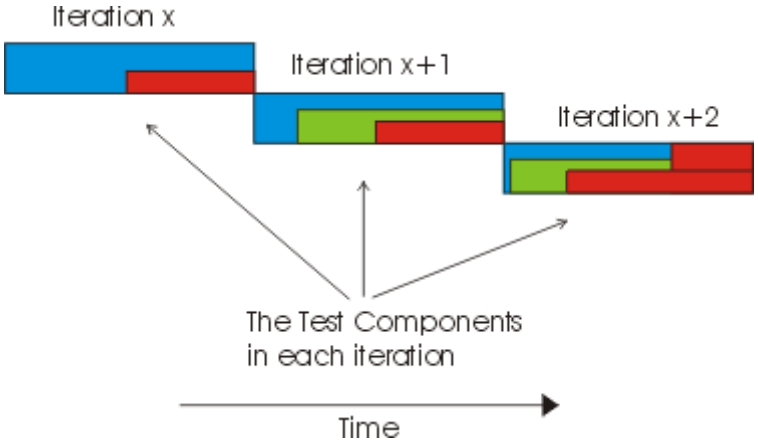
- Inicio: El desarrollo del prototipo exploratorio de demostración; no requiere la elaboración de pruebas.
- Elaboración: Probar los componentes ejecutables que se han implementado y que deben corresponderse con la arquitectura básica de la aplicación.
- Construcción: Desarrollar los casos de prueba y procedimientos de prueba para hacerlos.

- Transición: El producto en su entorno de operación por lo que es probado por usuarios reales.

3.4.1 Ciclo de vida de Prueba

El software es refinado a través de iteraciones en el ciclo de vida. El ciclo de vida de prueba se beneficia siguiendo un proceso iterativo equivalente. En cada iteración el equipo de desarrollo produce uno o más builds, cada builds es un candidato potencial para probar. Los objetivos del equipo de desarrollo difieren de una iteración a otra. El equipo de prueba estructura su prueba de acuerdo a los objetivos de la iteración.

En una iteración X, se puede implementar y ejecutar determinadas pruebas. Algunas de estas pruebas son conservadas y acumuladas, las cuales son usadas para pruebas de regresión. Cualquiera de las pruebas desarrolladas en la iteración X son candidatas para pruebas de X+1, cuando hay pruebas que son repetidas varias veces, vale la pena considerar automatizarlas.



3.4.2 Flujo de trabajo de la disciplina de Prueba

Los problemas encontrados durante una iteración pueden ser solucionados dentro de la misma iteración o pospuesto hasta próximas iteraciones.

Una de las tareas principales para el equipo de prueba y los administradores de proyecto es medir cómo la iteración finalizó, verificando que los objetivos de la misma expuestos en el Plan de Iteración están cumplidos.

3.4.3 Niveles de Prueba

La Prueba es aplicada para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo. Se distinguen los siguientes niveles de pruebas:

Nivel de Unidad

- Enfocada al código fuente de los componentes.
- Para verificar todos los flujos de control.
- Primero pasa por la revisión del programador.

Nivel de Integración

- Prueba los componentes combinados para ejecutar un CU.
- Para verificar y descubrir errores o incompletitud en las especificaciones de las interfaces de las clases.

Nivel de Sistema

- Prueba el software funcionando como un todo.
- Aceptable para cuando el software se encuentra en la Fase de Construcción.

Nivel de Aceptación

- Prueba final antes del despliegue del sistema.
- Generalmente lo realizan los usuarios finales.
- A veces se llama “Prueba Piloto”.

3.4.4 Tipos de Prueba

Cada tipo de prueba tiene un objetivo específico y una técnica que lo soporte.

Tipo de Prueba adecuada al Nivel de Unidad

La prueba de caja negra se refiere a las pruebas que se llevan a cabo sobre la interfaz del software. La prueba de caja blanca del software en esta se comprueban los caminos lógicos del software proponiendo casos de prueba que se ejerciten conjuntos específicos de condiciones y/o bucles.

Tipo de Prueba adecuada al Nivel de Integración

Prueba basada en hilos: Integrar las clases necesarias para cumplir eventos sistema.

Prueba basada en uso: Integrar las clases independientes primero, y luego las dependientes

3.4.5 Métodos de Prueba Basados en Caja Blanca

Prueba del camino básico: Esta prueba permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución.

Prueba de condición: Es un método de diseño de casos de prueba que ejercita las condiciones lógicas contenidas en el módulo de un programa.

Prueba de flujo de datos: Se selecciona caminos de prueba de un programa de acuerdo con la ubicación de las definiciones y los usos de las variables del programa.

Prueba de bucles: Es una técnica de prueba de caja blanca que se centra exclusivamente en la validez de las construcciones de bucles.

3.4.6 Estrategia de Prueba

La estrategia de prueba describe el enfoque y los objetivos generales de las actividades de prueba. Incluye los niveles de prueba (unidad, integración, etc.) a ser diseccionados y el tipo de prueba a ser ejecutadas (funcional, stress, etc.).

La estrategia define:

- Técnicas de pruebas (manual o automática) y herramientas a ser usadas.
- Qué criterios de éxitos y culminación de la prueba serán usados.
- Consideraciones especiales afectadas por requerimientos de recursos o que tengan implicaciones en la planificación.

Los tipos de pruebas se enfocan en dependencia del número de iteraciones, el tamaño de la iteración y el tipo de proyecto que se está probando.

En los programas Orientados a Objetos la estrategia y las tácticas de las pruebas cambian. Para probar los sistemas OO adecuadamente, se deben hacer tres cosas:

1. La definición de las pruebas debe ampliarse para incluir técnicas de detección de errores aplicados a los modelos de Análisis y Diseño Orientado a Objetos.
2. La estrategia para las pruebas de unidad e integración deben cambiar significativamente.
3. El diseño de casos de prueba[15] debe tener en cuenta las características propias del software orientado a objetos.

3.4.7 Diseño de Casos de Prueba

Un caso de prueba es:

- Un conjunto de entradas de pruebas, condiciones de ejecución y resultados esperados desarrollados para cumplir un objetivo en particular ó una función esperada.
- La entidad más simple que siempre es ejecutada como una unidad, desde el comienzo hasta el final.

Los casos de pruebas deben verificar:

- Si el producto satisface los requerimientos del usuario, tal y como se describe en las especificación de los requerimientos.

[15] Ver en Anexo 10 implicaciones de los conceptos de OO para el diseño de los casos de prueba

- Si el producto se comporta como se desea, tal y como se describe en las especificaciones funcionales del diseño.

La ejecución de un caso de prueba de un programa P:

1. direccionará (cubrirá) algunos requerimientos de P.
2. utilizará (cubrirá) algunas partes de la funcionalidad de P.
3. ejecutará (cubrirá) algunas partes de la lógica interna de P.

Las bases para una prueba es el material fuente que suministra el estímulo para la prueba.

1. Las pruebas para los requerimientos están basadas en los documentos de los requerimientos.
2. Las pruebas para las funciones están basadas en las especificaciones funcionales del diseño.
3. Las pruebas para la lógica interna están basadas en las especificaciones internas del diseño o el código.

Capitulo 4 Fase de Construcción y Transición

4.1 Introducción

En este capítulo se describen las fases de Construcción y Transición de RUP.

4.2 Fase de Construcción

El equipo que trabaja en la Fase de Construcción a partir de una línea base de la arquitectura ejecutable y trabajando a través de una serie de iteraciones e incrementos, desarrolla un producto software listo para su operación inicial en el entorno del usuario. Para ello, detalla los Casos de Usos y escenarios restantes, modifica si es necesaria la descripción de la arquitectura y continúa los flujos de trabajo a través de iteraciones adicionales, dejando cerrados los modelos de análisis, diseño e implementación. Además, integra los subsistemas y los prueba, e integra todo el sistema y lo prueba.

A medida que el proyecto pasa de la Fase de Elaboración a la de Construcción se produce un cambio de enfoque. Mientras que las fases de Inicio y Elaboración podrían ser consideradas como investigación, la fase de Construcción es análoga al desarrollo. El énfasis se traslada a la construcción de un sistema o producto dentro de unos parámetros de costo, esfuerzo y agenda. [7]

4.2.1 Flujo de trabajo de Requerimientos

Administrar requerimientos cambiantes: El propósito de esta actividad es evaluar el impacto de los cambios en los requerimientos y gestiona el impacto negativo de los cambios aprobados a ser ejecutados.

4.2.2 Flujo de trabajo de Análisis y Diseño

Diseñar componentes: El propósito de esta actividad es refinar el diseño del sistema.

Diseñar la base de datos: El propósito de esta actividad es identificar las clases de diseño que son persistentes en una base de datos y diseñar la estructura correspondiente de la base de datos.

Refinar la Arquitectura: El propósito de esta actividad es completar la Arquitectura para una iteración y actualizarla en caso de cambios.

4.2.3 Flujo de trabajo de Implementación

Plan de Integración: El propósito de esta actividad es desarrollar el Plan de Integración del sistema para la actual iteración.

Implementar componentes: El propósito de esta actividad es completar, iteración tras iteración, la implementación de cada uno de los componentes a fin de entregarlos a medida que se van implementando para ser integrados.

Integrar cada subsistema: El propósito de esta actividad es integrar cada componente actualizado de múltiples implementadores para crear una nueva versión consistente de un subsistema de implementación.

Integrar sistema: El propósito de esta actividad es integrar subsistemas de implementación para crear una nueva versión del sistema completo.

4.2.4 Flujo de trabajo de Prueba

Definir misión de evaluación: El propósito de esta actividad es identificar el enfoque apropiado del esfuerzo de la prueba para la iteración, y para llegar a un acuerdo con los interesados en las metas correspondientes que dirigirán el esfuerzo de la prueba.

Verificar método de prueba: El propósito de esta actividad es demostrar que las técnicas de pruebas utilizadas facilitarán el esfuerzo de la prueba planeado. El objetivo es verificar por demostración que esto trabajará para producir los resultados exactos y será apropiado para los recursos disponibles.

Validar estabilidad de construcción: El propósito de esta actividad es validar que la construcción es bastante estable para la prueba detallada y el esfuerzo al comienzo de la evaluación. A este trabajo se le llama “prueba de humo, prueba de comprobación de construcción, prueba de regresión de construcción, chequeo de sanidad o aceptación por probar”. Este trabajo ayuda a prevenir los recursos de las pruebas gastados en un inútil e infructífero esfuerzo de prueba.

Probar y evaluar: El propósito de esta actividad es lograr el alcance apropiado y la profundidad del esfuerzo de la prueba para permitir una evaluación suficiente de los elementos que son destinados para la prueba - donde la evaluación suficiente es controlada por los motivadores de la prueba actual y la misión de evaluación.

Lograr una misión aceptable: El propósito de esta actividad es entregar un resultado útil de la evaluación al interesado del esfuerzo de prueba — donde el resultado útil de la evaluación se evalúa en términos de la Misión de la Evaluación. En la mayoría de los casos significará la convergencia de los esfuerzos en ayudar que el proyecto logre los objetivos del Plan de Iteración que se aplican al ciclo de la prueba actual.

Mejorar pruebas: El propósito de esta actividad es mantener y mejorar las pruebas. Esto es sobre todo importante si la intención es rehusar los recursos desarrollados en el ciclo de la prueba actual en los ciclos de la prueba subsecuentes.

4.3 Fase de Transición

La fase de Transición comienza a menudo con la entrega de una versión beta del sistema, es decir, la organización distribuye un producto software capaz ya de un funcionamiento de inicial a una muestra representativa de la comunidad de usuarios. El funcionamiento del producto en el entorno de los usuarios es frecuentemente una prueba del estado de desarrollo del producto más severa que el funcionamiento en el entorno del que lo desarrolla. [11]

Objetivos y tareas involucradas en esta fase:

- Instalación del software en el entorno final de trabajo, realizando instalaciones progresivas y pruebas.
- Capacitación de los usuarios con la nueva herramienta.
- Conversión e importación de datos anteriores al nuevo sistema.
- Ajuste del software y la organización.
- Medición de performance de la herramienta y del esquema organizacional.
- Pruebas de estrés sobre las redes y equipamiento, verificación de los planes de contingencia.

Cada una de estas etapas es desarrollada mediante el ciclo de iteraciones, la cual consiste en reproducir el ciclo de vida en cascada a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes.

La fase de Transición finaliza con la entrega de del producto final. Sin embargo, antes de que el equipo del proyecto abandone el proyecto, los líderes del equipo llevan a cabo un estudio del sistema con los siguientes objetivos:

- Encontrar, discutir, evaluar y registrar las lecciones aprendidas para referencias futuras.
- Registrar asuntos útiles para la entrega de o versión siguiente.

Capítulo 5 Adaptar RUP para proyectos con equipos de trabajo pequeños.

5.1 Introducción

Generalmente RUP es utilizado para proyectos grandes, en los que se demanda la presencia de muchos integrantes debido al alto grado de formalidad que presenta. Donde la coordinación y comunicación requeridas para la colaboración de estos; agregan otro nivelado de complejidad a un proceso creativo ya complejo. Otro problema radica en que muchas veces no se cuenta con la cantidad de trabajadores que necesita RUP por definición.

En este capítulo se contextualiza RUP en proyectos con equipos de trabajo de pocos miembros. En estos proyectos el nivel de formalidad estará en correspondencia con el número de las personas en el proyecto y el nivel de complejidad. También se realiza una propuesta de RUP adaptado según el tipo de aplicación.

Para esto se tuvieron en cuenta criterios de algunos expertos racionales de RUP como por ejemplo una entrevista a " Gary Pollice". [11]

(…) RUP es muy flexible (…) usted puede crear sus propias pautas, adapte elementos del proceso para encontrar las necesidades de su proyecto (…). Usted adaptará RUP a sus necesidades. Una parte importante de un caso de desarrollo es que explica las responsabilidades de cada papel diferente en el proyecto. Con un equipo pequeño cada persona juega típicamente más de un papel, para esto es importante definir las responsabilidades cuidadosamente. (…). Usted puede mezclar los niveles de formalidad, si usted está en un equipo de tres-personas las comunicaciones y la parte del proyecto puede ser muy informal.

5.2 Utilización de RUP en nuestro territorio

Para conocer el grado de utilización de RUP en nuestro territorio, fue necesario aplicar una encuesta; para obtener información cualitativa y cuantitativa. En el Anexo 1 se puede ver el modelo de la encuesta aplicada.

Las preguntas de la encuesta fueron elaboradas para determinar cómo las empresas de software en nuestro territorio utilizan RUP y si cumple con las expectativas; principalmente cuando el equipo de desarrollo es reducido.

Para el diseño de la encuesta se combinaron los dos tipos de preguntas fundamentales; abierta y cerradas. Las preguntas cerradas se realizaron con el objetivo de adquirir la información cuantitativa con respecto al tema y las preguntas abiertas con el fin de saber la opinión del encuestado.

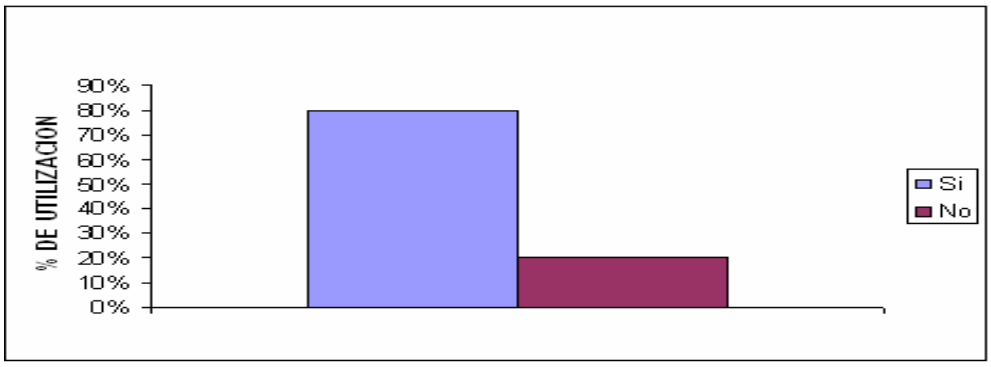
5.3 Análisis de la Encuesta

Las encuestas se efectuaron en 3 empresas desarrolladoras de software de nuestro territorio; Desoft Holguín, Serconi Moa y en la Empresa Mecánica del Níquel, también en 2 centros universitarios Instituto Superior Minero Metalúrgico de Moa ¨Dr. Antonio Núñez Jiménez¨ y a la Universidad de Holguín ¨Oscar Lucero Moya¨.

Haciendo un análisis por preguntas se puede decir que:

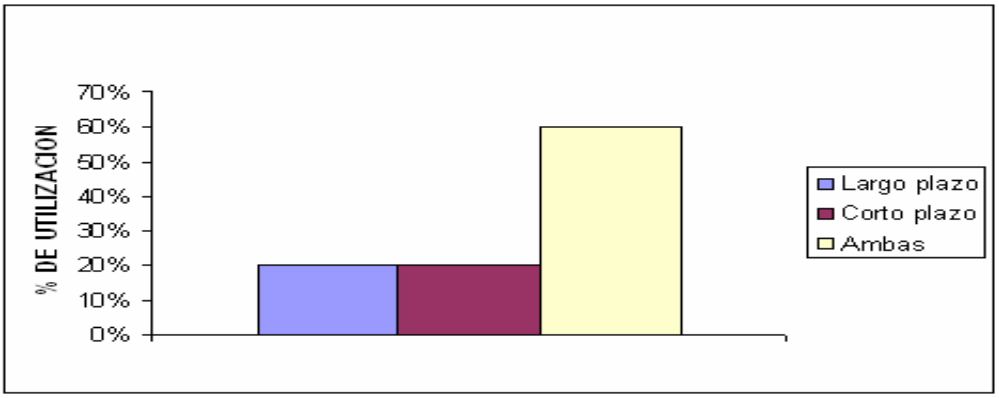
Pregunta 1:

Esta pregunta arrojó que de las 25 personas entrevistadas el 80% usan RUP para construir software. Por lo que se puede concluir en que la mayoría de estas empresas utilizan RUP como metodología en el proceso de desarrollo de su proyecto.



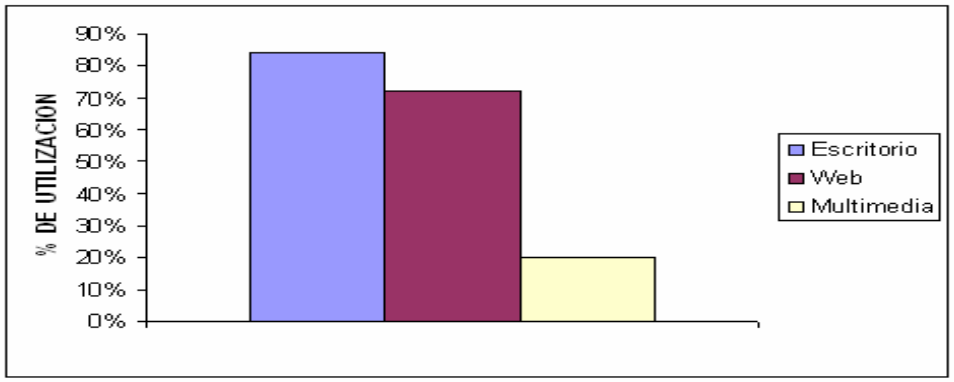
Pregunta 2:

El 20% utiliza RUP para realizar proyectos a largo plazo, el 20% a corto plazo y el 60% para ambos casos.



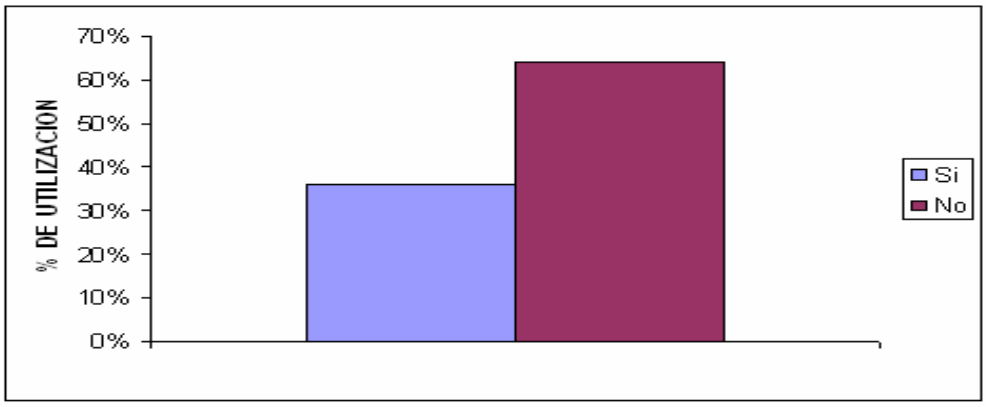
Pregunta 3:

El 84% usa RUP para aplicaciones de escritorio, el 72% en aplicaciones web y el 20% en aplicaciones multimedia.



Pregunta 4:

El 64% de los encuestados confirmaron que RUP no es eficiente en proyectos con equipos de trabajo pequeños.



Pregunta 5:

En esta pregunta el encuestado sugiere lo que considera necesario para lograr la eficiencia de RUP en proyectos con equipos de trabajo pequeños, estos sugieren que hay que distinguir entre lo necesario y lo que no lo es, para evitar retrasos en la entrega de los productos; no se deben utilizar demasiados artefactos y sobre todo tener bien definidos estos entregables según el tipo de proyecto a desarrollar, también proponen que se eliminen roles, y algunos no tenían conocimiento a cerca del tema.

De esta forma se demuestra la necesidad de adaptar RUP para proyectos con grupos de trabajo reducidos, ya que estos fracasan debido a que esta metodología requiere de mucho personal, con los que generalmente; dada nuestra realidad no se cuentan, y la importancia de definir y documentar los entregables según el tipo de proyecto.

5.4 Propuesta de RUP para proyectos con equipos de trabajo pequeños.

Haciendo una valoración de los roles de RUP, las actividades que realizan en cada uno de sus flujos de trabajo, y los artefactos que se generan; se puede concluir en que los mismos se pueden integrar obteniéndose tres roles fundamentales: Analista, Desarrollador y Tester. A continuación se describen cada uno con sus actividades y los artefactos que se generan durante el proyecto.

5.4.1 FT Elicitación de Requisitos

En este flujo se obtiene una visión de la organización, que permite definir los objetivos y riesgos del proyecto, procesos, actores, identificar y detallar los casos de uso, representar los requisitos funcionales como casos de uso y recoger los requisitos no funcionales relacionados.

Analista

- Decide cuáles son los actores, los procesos del negocio y la relaciones entre ellos y cuáles son las reglas del negocio a tener en cuenta.
- Identifica a las entidades del negocio, trabajadores del negocio y sus relaciones.
- Dirige y coordina el proceso de Modelamiento del Negocio.
- Revisa formalmente el modelo de CUN.
- Responsable de la arquitectura del negocio.
- Define cuáles son los requerimientos en la automatización.
- Define los alcances del sistema.
- Identifica las demandas de los stakeholders.
- Desarrolla el Plan de Gestión de Requerimientos.
- Captura los requisitos del sistema (funcionales, de usabilidad, de capacidad, etc.).
- Identifica Actores y Casos de Uso del sistema.
- Describe detalladamente cada caso de uso de acuerdo a las funcionalidades que engloba.
- Estructura el Modelo de Casos de Uso del Sistema.
- Definición del prototipo no funcional.
- Define los diferentes artefactos según la metodología a utilizar. Fundamentar la elección de la metodología.
- Definir estrategia de captura de requisitos. Técnicas, métodos y plantillas a utilizar.

Artefactos

- Documento Visión (Descripción del Negocio, objetivos del proyecto, usuarios del sistema, requerimientos funcionales y no funcionales)
- Reglas del Negocio
- Modelo de CUN
- Realización de cada CUN
- Diagramas de Actividades

Si el Negocio tiene un bajo nivel de estructuración.

- Modelo de Dominio
- Modelo de Casos de Uso del Sistema

Desarrollador

- Describe la vista de la arquitectura del modelo de casos de uso.
- Define la prioridad de cada caso de uso para decidir en qué iteración será desarrollado cada uno.

1. Combinar algunos de los resultados.
2. Definición de las herramientas a utilizar para el desarrollo.
3. Definición de la estrategia de desarrollo.
4. Definición de la configuración de los puestos de trabajo según los roles.
5. Estrategia de comunicación para lograr la integración de las diferentes partes.
6. Fundamentación de los patrones a utilizar, así como su aplicación.

Artefactos

- Descripción de la arquitectura (vista del modelo de casos de uso)
- Prototipo de interfaz usuario

Se propone que al finalizar este flujo se contacte con el usuario para validar los requisitos y los prototipos de interfaz del usuario.

5.4.2 FT Análisis y Diseño

FT Análisis

El análisis permite obtener un enfoque del sistema, se preocupa fundamentalmente por los requisitos funcionales. Los cuales son refinados y estructurados. El resultado de este flujo es modelo de análisis; este modelo ayuda en el establecimiento de la arquitectura candidata.

Desarrollador

- Define las responsabilidades, las operaciones, los atributos y las relaciones de una o varias clases.
- Responsable de la integridad del modelo de análisis y de la arquitectura del modelo de análisis.

Artefactos

- Modelo de Análisis
- Clases de Análisis
- Descripción de la arquitectura (vista del modelo de análisis)

FT Diseño

En el Diseño se perfecciona el Análisis; se realiza un modelo de diseño a partir de la arquitectura candidata, este flujo se preocupa porque el sistema cumpla con los objetivos trazados (requisitos no funcionales), se diseñan clases, subsistemas y el modelo de despliegue.

Desarrollador

- Responsable del modelo de diseño, modelo de despliegue, descripción de la arquitectura.
- Responsable de los paquetes de diseño, o diseño del subsistema.

Si BD:

Definir las tablas, índices, vistas, constraints, triggers, y otros objetos específicos de la BD para almacenar, recuperar y eliminar objetos persistentes.

- Artefactos de la metodología que describen la base de datos.
- Integración de la solución de los diferentes módulos.
- Fundamentación de la definición de un sistema gestor de base de datos.
- Definición de la estrategia y la tecnología de acceso a datos a utilizar.
- Definición de los modelos de despliegue de la base de datos.
- Definición de la implementación de la réplica en función de las necesidades de la solución.
- Definición de las herramientas a utilizar para el diseño de la base de datos y para el rendimiento.

Artefactos

- Modelo de Diseño
- Clase del Diseño
- Descripción de la arquitectura (vista del modelo de diseño)
- Subsistema de Diseño
- Realización de Casos de Uso del Diseño

- Modelo de Despliegue

Si tiene Base de Datos:

- Diseño de la Base de Datos

Desarrollador

- Diagrama de Clases Persistentes
- Modelo de Datos (Se genera este entregable si existe Base de Datos)

En este flujo los tres trabajadores actuarían como un solo rol, se integran en el rol de Desarrollador los cuales se dividen las actividades, primeramente los casos de usos prioritarios y después los secundarios.

Los miembros del equipo, al mismo tiempo que se realizan las actividades del análisis y diseño, van considerando que pruebas se requerirán por lo que van desarrollando planes temporales de pruebas.

5.4.3. FT Implementación

Apoyándose en la vista de la arquitectura del modelo de diseño y la vista de la arquitectura del modelo despliegue se identifican los componentes para implementar los subsistemas, se implementan subsistemas y clases, se integra el sistema.

Desarrollador

- Estructurar el modelo de Implementación.
- Determina como serán ajustadas las clases al ambiente de implementación.
- Planificar la Integración al Sistema.
- Implementación de Componentes.

Resultados:

1. Estándares de codificación.
2. Componente o elemento implementado.
3. Análisis crítico de la tecnología o lenguaje utilizado.
4. Análisis de Integración con otros componentes o partes del sistema.

Descripción de los elementos que resultan interfaz de comunicación con el resto de las partes del sistema.

- Análisis de algoritmos:

1. Conocidos
2. Análisis de eficiencia. Demostrar el máximo de eficiencia.
3. Nuevos algoritmos:

Demostración y análisis de eficiencia. Carencia del algoritmo necesitado.

- Pruebas (caja blanca):

1. Estrategias de pruebas.
2. Diseño de pruebas.
3. Ejecución (profiling) y resultados de las pruebas. Análisis de resultados.

• Documentos de diseño. La especificación máxima de la documentación de diseño, con las decisiones de implementación.

• Especificaciones de instalación en caso de que el componente llevase instalación.

• Utilización de patrones de diseño.

• Responsable de la integridad del Modelo de Implementación de acuerdo a lo descrito en el Modelo de Diseño.

• Encargado de la arquitectura del Modelo de Implementación.

• Planifica la secuencia de construcciones necesarias en cada iteración y la integración de cada construcción cuando sus partes han sido implementadas. Esta planificación da lugar a un plan de integración de construcciones.

Artefactos

- Modelo de Implementación
- Plan de Integración
- Componente
- Interfaz
- Subsistema de implementación
- Diagrama de Componentes

En este flujo también los tres trabajadores se integran, actuando como Desarrollador los cuales se dividen las actividades.

Los miembros del equipo, al mismo tiempo que se realizan las actividades de la implementación, tienen en cuenta que pruebas se necesitaran por lo que van desarrollando planes temporales de pruebas.

5.4.4. FT Prueba

En este flujo se planifican y se diseñan las pruebas según los objetivos a evaluar. Se realizan pruebas del sistema y de integración. Con los resultados que se obtienen se comprueba si se cumplió con los objetivos trazados.

Analista

- Es el responsable de identificar y definir las pruebas requeridas, monitorear el progreso de la prueba y el resultado en cada ciclo de prueba, evaluando la calidad total.

Desarrollador

- Responsable de definir el método de prueba y asegurar su implementación exitosa.
- Identificar técnicas apropiadas, herramientas e instrucciones para implementar las pruebas necesarias y encauzar los recursos correspondientes para las pruebas.
- Es el responsable durante las actividades principales de las pruebas, de la conducción de las pruebas necesarias y el registro del resultado de estas.
- Es el responsable de definir el método de prueba y asegurar su implementación exitosa.

Tester

- Conducir las pruebas necesarias y registrar los resultados de aquello que prueba.

Esto cubre:

1. Identificar la implementación más apropiada para una prueba efectuada.
2. Implementar pruebas individuales.
3. Crear y ejecutar las pruebas.
4. Registrar los resultados verificar la ejecución de la prueba.
5. Analizar y recuperar errores de ejecución.

Artefactos

Analista, Desarrollador, Tester

- El plan de Pruebas
- La estrategia de prueba
- Los casos de pruebas
- Los Procedimientos de prueba
- El listado de datos de prueba
- El resumen de la evaluación de las pruebas

Para este flujo en caso de tener que prescindir del rol de Tester; las actividades se pueden dividir entre el Analista y el Desarrollador.

5.5. RUP adaptado según el tipo de aplicación para grupos de trabajo de pocos miembros

A continuación se hace una propuesta de RUP adaptado según el tipo de aplicación, esta servirá para equipos pequeños de desarrollo de software, los artefactos que se generan en la misma se definen para cada tipo de aplicación:

- Propuesta General

- Aplicaciones de Escritorio
- Aplicaciones Web
- Aplicaciones Multimedia

5.5.1. Propuesta General

- Documento Visión (Descripción del Negocio, objetivos del proyecto, usuarios del sistema, requerimientos funcionales y no funcionales)
- Reglas del Negocio
- Diagrama de Caso de Uso del Negocio
- Diagrama de Actividades

En caso de baja estructuración:

- Modelo de Dominio
- Diagrama de Caso de Uso del Sistema
- Prototipo de interfaz usuario
- Clases de Análisis
- Clases del Diseño
- Realización de Casos de Uso del Diseño
- Diseño de la BD
- Diagramas de Secuencia
- Modelo de Despliegue
- Modelo de Componentes
- Plan de Pruebas
- Estrategia de Pruebas
- Casos de Pruebas
- Procedimientos de Pruebas
- Listado de Datos de Pruebas
- Resumen de la Evaluación

5.5.2 Propuesta para Aplicaciones de Escritorio

Los mismos artefactos que en la propuesta para cualquier tipo de aplicación.

En caso de no tener base de datos se eliminan los artefactos correspondientes a la BD.

5.5.3 Propuesta para Aplicaciones Web

Los mismos artefactos que en la propuesta general e incluyendo:

- Diagrama de Clases Web
- Mapa de Navegación
- Diagrama de Presentación

En caso de no tener base de datos se eliminan los artefactos correspondientes a la BD.

5.5.4. Propuesta para Aplicaciones Multimedia

En caso de una Multimedia usando como lenguaje de modelación OMMA-L los artefactos son los siguientes:

Los mismos artefactos que en la propuesta general e incluyendo:

- Diagrama de Presentación

En caso de no tener base de datos se eliminan los artefactos correspondientes a la BD.

Capítulo 6 Ejemplificar RUP adaptado en diferentes tipos de proyectos de software.

6.1 Introducción

En este capítulo se ejemplifica la propuesta de RUP adaptado en los diferentes tipos de aplicaciones, lo que facilitará la comprensión de dicha propuesta.

6.2 Ejemplo para Aplicaciones de Escritorio

Problema

Se necesita automatizar la gestión de control de la entrega de medios (básicos, de uso y de aseo personal) a los estudiantes becados del ISMM. La acción de gestionar el control de los medios asignados a los estudiantes becados del ISMM presenta deficiencias en las entregas, debido a que la información sobre los estudiantes becados se realiza manualmente, lo que trae como consecuencia demoras en las actualizaciones del almacén. El control de los medios existentes en el almacén, así como los entregados/recibidos de los estudiantes becados; se hacen de forma manual esto representa riesgos para los bienes materiales del ISMM y del país. Por lo que se propone digitalizar la información sobre los medios.

El Sistema tendrá un administrador (jefe de beca) que será el encargado de abastecer y chequear los medios existentes en el almacén. En la aplicación se mostraran todos los medios (productos), así como la cantidad que se encuentren disponibles para la entrega a los estudiantes becados. Cada estudiante adquiere estos medios a través de su carnet de becado para identificarse, luego de haber recibido los mismos tendrá que firmar como confirmación de lo que ha recibido, en caso de que lo adquirido sea un medio en uso en la tarjeta de datos del becado, si es medio básico en el acta de responsabilidad, y en caso de aseo personal en el registro de control. Deberá chequearse que el estudiante no este sancionado o de baja, en cualquiera de estos casos no se le entrega ningún medio.

Cada estudiante becado tiene una tarjeta de datos en el almacén donde se registran sus datos (nombre, apellidos, edad, fecha de nacimiento, lugar de nacimiento, dirección particular, ubicación (# de cuarto), año, especialidad, facultad) y los datos de los medios (nombre del producto, tipo de medio (aseo personal, básico. en uso), cantidad, fecha de entrega, nombre del que entrega, nombre del que recibe).

Una vez que este trabajo este terminado el personal autorizado a trabajar con la aplicación no

tendrá la necesidad de llevar el control de los productos de forma manual.

Se desea que el sistema funcione cumpliendo los siguientes objetivos estratégicos trazados:

1. Automatizar la gestión de entrega de los diferentes medios a los estudiantes becados en el ISMM.
2. Contribuir al desarrollo tecnológico de la universidad.
3. Mantener informados a los empleados del almacén sobre los estudiantes que ya recogieron sus medios.
4. Realizar un sistema eficiente que responda a los intereses de los trabajadores del almacén.

Podemos concluir que lo que se requiere es optimizar el proceso de entrega de los medios a los estudiantes becados del centro, para brindar un servicio eficaz y eficiente.

Descripción del negocio

El estudiante becado acude al almacén, presentando su carnet, este solicita los medios que le corresponden, es atendido por la distribuidora que comprueba con los datos de su carnet en el listado de estudiantes si no está sancionado o de baja, porque en cualquiera de estos casos no se le entregan medios, en caso contrario el estudiante especifica el tipo de medio que quiere recoger, si el medio es básico lo localiza en el acta de responsabilidad, si es un medio en uso lo localiza en la tarjeta de datos, si es aseo personal en el registro de control, en cualquiera de estos documentos el estudiante firma como constancia de que recibió el medio, la distribuidora entrega los medios solicitados y recoge el documento, el estudiante se retira con sus medios.

Reglas del negocio

Regla 1: No se les puede entregar medios a estudiantes que no sean becados.

Regla 2: El estudiante debe identificarse con su respectivo carnet de becado.

Regla 3: Solo se entrega la cantidad de los medios de uso asignada.

Regla 4: No se les puede entregar medios a estudiantes dados de baja o sancionados.

Actores del negocio

Tabla 1. Descripción de los actores del negocio

Nombre del actor	Descripción
Estudiante Becado	Interviene en el proceso de recoger medios con el motivo de solicitar que estos le sean entregados.

Trabajadores del Negocio

Tabla 2. Descripción de los Trabajadores del Negocio

Nombre del trabajador	Descripción
Distribuidora	Se encarga de llenar la Tarjeta de Datos del Becado, el Registro de Control y notificar las Actas de Responsabilidad, verifica los datos de los becados en la lista de estudiantes.

Diagrama de casos de uso del negocio

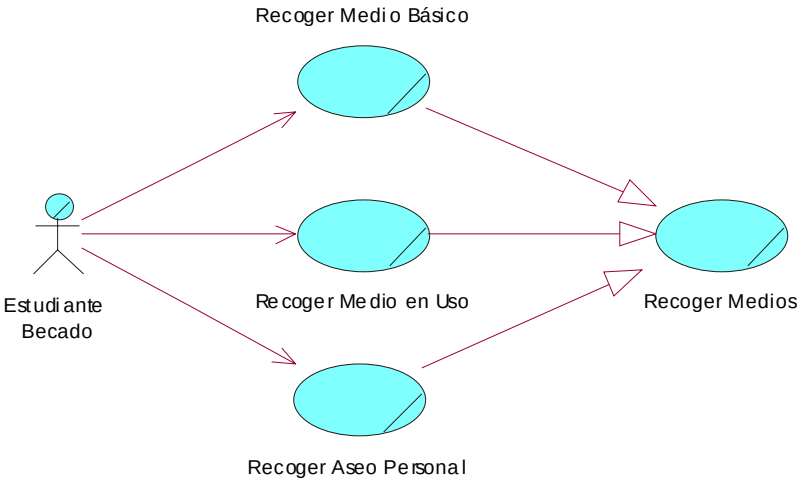


Figura 1. Diagrama de Casos de Uso del Negocio

Tabla 3. Descripción del Caso de Uso <*Recoger Medios*>

Caso de Uso:	Recoger Medios
Actores:	Estudiante Becado (inicia)
Trabajadores:	Distribuidora
Resumen:	El caso de uso se inicia cuando un estudiante llega al almacén y solicita los medios que le corresponden, este es atendido por la distribuidora que con los datos del mismo comprueba en el listado de estudiantes que el estudiante no esté sancionado o de baja, verifica el tipo de medio solicitado (en uso, básico o aseo personal) lo localiza respectivamente en su tarjeta de datos del becado, acta de responsabilidad o registro de control y le entrega los medios asignados.

	Finalizando así el caso de uso.	
Precondiciones:	El estudiante debe llevar el carnet de estudiante.	
Flujo Normal de Eventos		
Acción del Actor		Respuesta del Negocio
1. El estudiante becado acude al almacén presentando el carnet de becado.		1.1. La distribuidora del almacén comprueba en el listado de estudiantes que el estudiante no esté sancionado o de baja. 1.2. Verifica el tipo de medio solicitado. 1.2.1. Si es un medio básico lo localiza en el acta de responsabilidad. 1.2.2. Si es un medio en uso lo localiza en su tarjeta de datos. 1.2.3. Si es aseo personal lo localiza en el registro de control. 1.3. La distribuidora del almacén entrega medio solicitado al estudiante becado.
2. El estudiante becado recoge los medios y firma en el documento correspondiente al tipo de medio. Retirándose del almacén.		2.1. La distribuidora del almacén recoge el documento correspondiente (acta de responsabilidad, tarjeta de datos, registro de control).
Flujos Alternos		
Acción del Actor		Respuesta del Negocio
Poscondiciones	El estudiante se retira con sus medios.	

Diagrama de Actividades

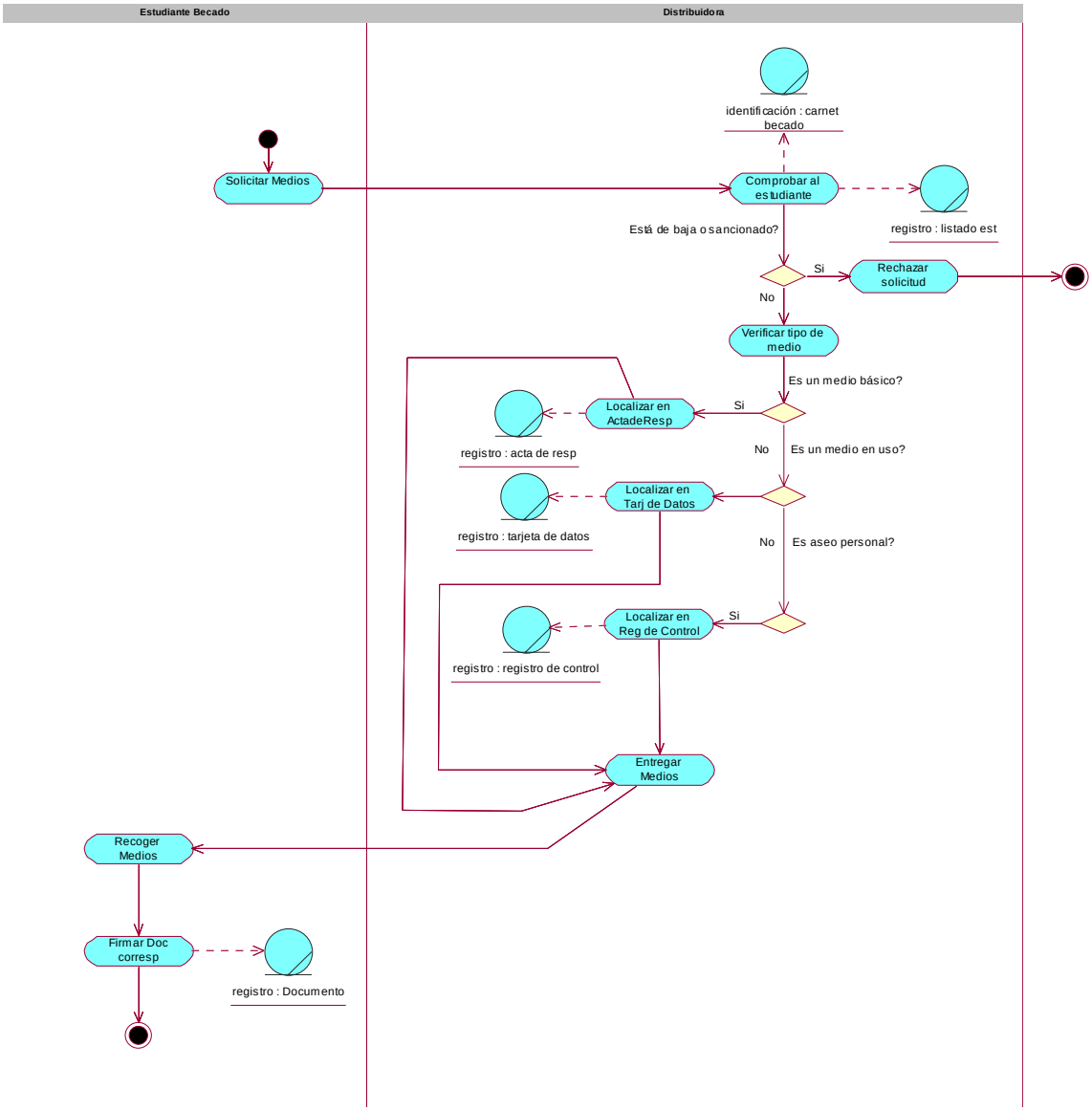


Figura 2. Diagrama de Actividades

Requerimientos Funcionales y no Funcionales del Sistema

Requisitos Funcionales

La respuesta será en función solamente de la entrega de medios, entonces teniendo en cuenta los objetivos de los futuros usuarios del sistema y la descripción de cómo debe funcionar el mismo, se pueden inferir los requerimientos funcionales siguientes:

El sistema debe ser capaz de:

R 1-Gestionar usuario: Permitir a los usuarios acceder a la información que le corresponde.

R1.1- Comparar Usuario y contraseña con los usuarios del sistema.

R1.2- Asignar privilegios.

R 2- Controlar Existencia de medios.

R2.1- Registrar los tipos de medios adquiridos.

R 2.2- Actualizar datos de los medios adquiridos.

R 2.3- Eliminar medios.

R 2.4- Listar medios existentes en el almacén.

R 3- Controlar tenencia de medios.

R 3.1- Buscar al estudiante por su carnet de becado.

R 3.2- Comprobar medios pendientes.

R 3.3- Mostrar Lista de medios disponibles.

R 3.4- Registrar la entrega.

R 4- Mostrar presentación del sistema.

Requisitos no Funcionales

- Apariencia o interfaz externa

Del sistema las partes que interactuarán con el usuario tendrán una interfaz amistosa que favorezcan a la configuración del sistema.

Para esta aplicación no se cuenta con una interfaz visual específica, no son necesarias hacer caracterizaciones de la misma.

- Usabilidad

El sistema debe ser concebido para dar facilidad de uso a personas sin experiencia previa.

- Rendimiento

El tiempo de respuesta dependerá de la programación, del SO, de la plataforma de hardware donde corre el sistema.

- Políticos-culturales

La herramienta propuesta deberá responder a los intereses de la Constitución de la República de Cuba, asimismo no existirán prioridades en el servicio según el nivel social, cultural o étnico.

- Confiabilidad

La información manejada por el sistema está protegida de acceso no autorizado y divulgación.

- Software y Hardware

Para lograr un buen funcionamiento de este sistema se necesita tener instalado el Microsoft Excel en una de sus versiones y una computadora con Microprocesador Pentium II o superior con más de 600 MHz y de espacio de disco se recomienda al menos 1 Gb de espacio libre. En caso de que se vaya a trabajar conectado a una Red industrial contar con una tarjeta de red.

- Restricciones en el diseño y la implementación.

Para su implementación se usó como lenguaje de programación Delphi 7, y para la base de datos Access. Para garantizar una mejor documentación del sistema, se utiliza para realizar el análisis y el diseño del sistema UML (Unified Modelling Language). Como herramienta de apoyo a este lenguaje de modelación se utiliza Rational Rose.

Prototipo

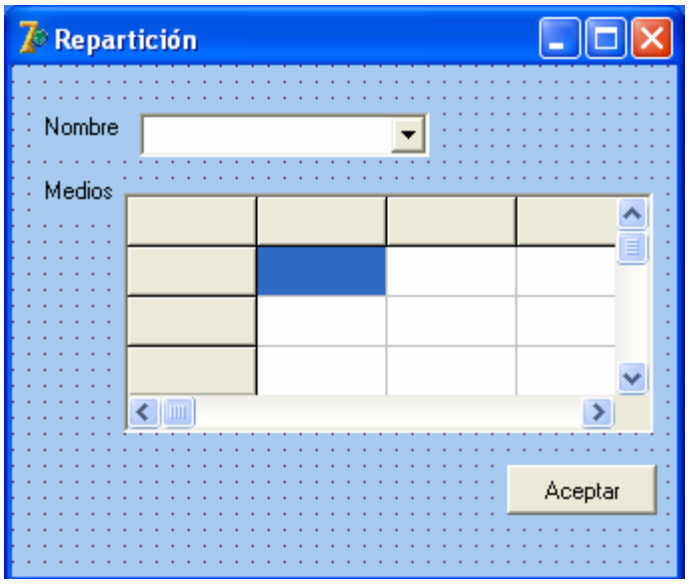


Figura 17. Prototipo de Interfaz

Análisis y Diseño del Sistema

Actores del sistema

Tabla 4. Definición de Actores del Sistema a automatizar

Nombre del actor	Descripción
Distribuidora	Es la encargada de llevar a cabo todo lo relacionado con la entrega de Medios a través de la aplicación. Actualiza la ficha de medios de cada estudiante en el sistema.
Administrador	Responsable de realizar todas las actualizaciones de existencia en el

	almacén; registra, elimina y modifica datos acerca de los medios en el sistema. Permite la autenticación de los usuarios al sistema.

Paquetes y sus Relaciones

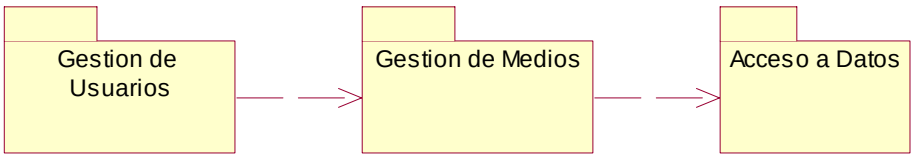


Figura 4. Diagrama de Paquetes

Diagrama de Casos de Uso del Sistema

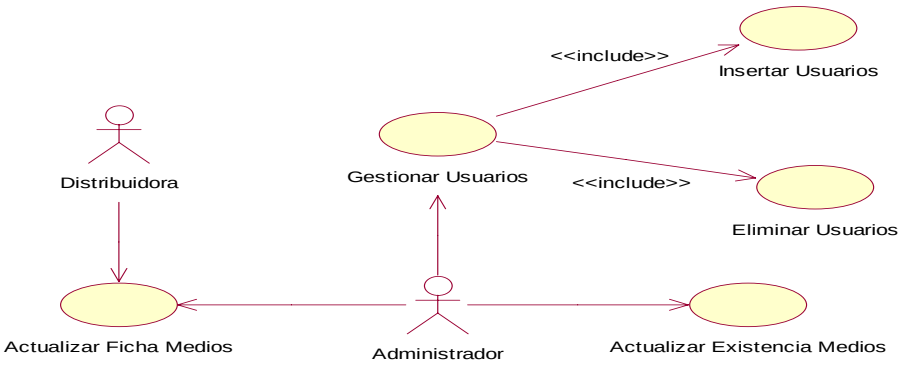


Figura 5. Diagrama del Caso de Uso del Sistema

Descripción de los Casos de Uso del Sistema

Tabla 5. Descripción del Caso de Uso del Sistema <Gestionar usuarios>

Nombre del Caso de Uso	Gestionar usuarios
Actores	Administrador del Sistema (inicia)
Propósito	Permitir a los usuarios acceder a la información que le corresponde.

Resumen	El Caso de Uso se inicia cuando el administrador introduce los datos que se le piden del usuario para que este pueda acceder a la aplicación, estos se verifican y finaliza dándole los permisos y autorizándole la entrada.
Referencias	R1
Precondiciones	
Poscondiciones	Se autorizan las funcionalidades según los privilegios.

Curso Normal de los Eventos	
Acciones del Actor	Respuesta del Sistema
1. El administrador entra Usuario y Contraseña.	1.1 El sistema encripta la contraseña. Busca el usuario y compara la contraseña. 1.2 En caso de ser correcto se le asignan los permisos.

Curso alternativo	
Acciones del Actor	Respuesta del Sistema
	1.3 En caso de no existir se envía un mensaje de aviso.

Prioridad	Crítico

Tabla 6. Descripción del Caso de Uso del Sistema <Actualizar Existencia de Medios>

Nombre del Caso de Uso	Actualizar Existencia de Medios
Actores	Administrador
Propósito	Permitir registrar, eliminar y modificar datos acerca de los Medios.
Resumen	Es aquí donde se registran, eliminan, y se modifican los datos de los Medios que entran en el almacén.
Referencias	R2

Precondiciones	Administrador del sistema ya autenticado.
Poscondiciones	Si se registra un medio básico, en uso o aseo personal se actualizan los datos o se eliminan.
Curso Normal de los Eventos	
Acciones del Actor	Respuesta del Sistema
El administrador del sistema necesita registrar, eliminar y modificar los datos de un Medio.	El sistema ejecuta algunas de las siguientes acciones: Si decide registrar cualquier tipo de medio, ir a la sección “Registrar Medio”. Si decide modificar las características de un Medio, ir a la sección "Actualizar Datos". Si decide eliminar un Medio, ir a la sección "Eliminar Medio”.
Sección “Registrar Medio”	
Acciones del Actor	Respuesta del Sistema
El Administrador del Sistema entra los datos del tipo de medio para realizar su registro en la aplicación.	2.1 El sistema verifica que los campos del código, nombre, el tipo de medio y la cantidad estén llenos. 2.2 El sistema verifica que este Medio no exista. 2.3 El Medio se almacena en el sistema. 2.4 Se muestra un mensaje donde se le informa al administrador que ya ha sido efectuado el registro del Medio y finaliza el caso de uso.
Curso alternativo	
	2.1 Se emite un mensaje para que llene los campos obligatorios. 2.3 Si el medio existe se muestra un mensaje informativo y finaliza el caso

	de uso.
Sección “Actualizar Datos”	
Acciones del Actor	Respuesta del Sistema
2. El administrador del sistema selecciona el medio a modificar.	2.1 El sistema brinda la posibilidad de modificar los datos.
3. El administrador del sistema realiza las actualizaciones deseadas.	3.1 Se verifica que los campos obligatorios estén llenos. 3.2 Se actualiza la información y finaliza el caso de uso.
Curso alternativo	
	3.1 Se emite un mensaje para que llene los campos obligatorios.
Sección “Eliminar Medio”	
Acciones del Actor	Respuesta del Sistema
2. El administrador del sistema selecciona el medio a eliminar.	2.1 El sistema pide confirmación. 2.2 El sistema elimina el medio.
Prioridad	Crítico

Tabla 7: Descripción del Caso de Uso del Sistema < Actualizar ficha de Medios>

Nombre del CU	Actualizar ficha de Medios
Actores	Distribuidora, Administrador
Propósito	Actualizar fichas de Medios por estudiante en caso de una entrega o recepción de medios.
Resumen	Es aquí donde se le entregan los medios al estudiante y se actualiza la cantidad real de medios en dependencia de lo que se le haya entregado.
Referencias	R3

Precondiciones	Distribuidora o Administrador del sistema ya autenticada
Poscondiciones	
Curso Normal de los Eventos	
Acciones del Actor	Respuesta del Sistema
1. La distribuidora o administrador del sistema entra el carnet del estudiante.	1.1 Se busca su ficha de medios. 1.2 Si tiene medios pendientes ir a sección "Devolución de medios" En caso contrario ir a sección "Entrega de medios"
Sección "Devolución de medios"	
Acciones del Actor	Respuesta del Sistema
	1.3 El sistema deshabilita la opción de Entregar de la distribuidora.
2. La distribuidora selecciona la opción de devolución.	2.1 El sistema actualiza la ficha del estudiante y así finaliza el caso de uso.
Sección "Entregar medios"	
Acciones del Actor	Respuesta del Sistema
2. La distribuidora selecciona la opción de Entregar medios.	2.1 El sistema habilita la ventana de entrega de los medios.
3. La distribuidora selecciona los medios a entregar.	3.1 El sistema registra la información y así finaliza el caso de uso.
Prioridad	Crítico

Diagrama de Clases del Análisis

Modelo de Análisis:

Para el Caso de uso Actualizar existencia de Medios:

1 Identificar las Clases del Análisis para el Caso de Uso

Clases Interfaz

• Registrar Medios (clase interfaz mediante la cual se adicionarán los datos necesarios para adicionar un nuevo medio)

- Modificar Medios (clase interfaz mediante la cual se actualizarán los datos de un medio existente en el almacén)
- Eliminar Medios (clase interfaz mediante la cual se eliminará un medio existente en el almacén)

Clases Entidad

- Ficha de Almacén de los Medios (es la principal plantilla que contiene los datos de un medio en el almacén)

Clases Control

- Gestor Registrar Medios (clase controladora que se encarga de hacer todos los procesos de crear, buscar y mostrar la entidad FichaMedios, desde la interfaz de Registrar Medios)
- Gestor de Medios (clase controladora que se encarga de hacer todos los procesos de buscar, mostrar, actualizar y eliminar en la entidad medio, desde las diferentes interfaces)

2 - Confeccionar Diagrama de Clases del Análisis:

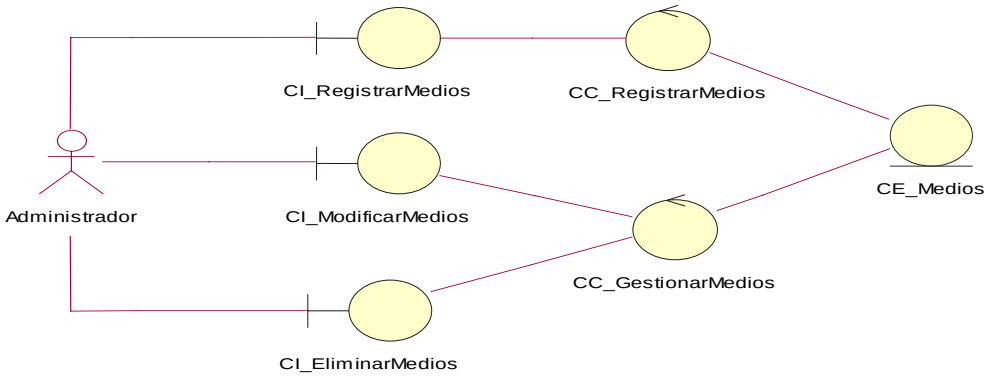


Figura 6. Diagrama de Clases del Análisis

Diagrama de Clases del Diseño

El diseño depende del lenguaje de programación utilizado, para la aplicación de escritorio se usó Delphi como lenguaje.

Diagrama de Clases del Paquete Gestión de Usuarios

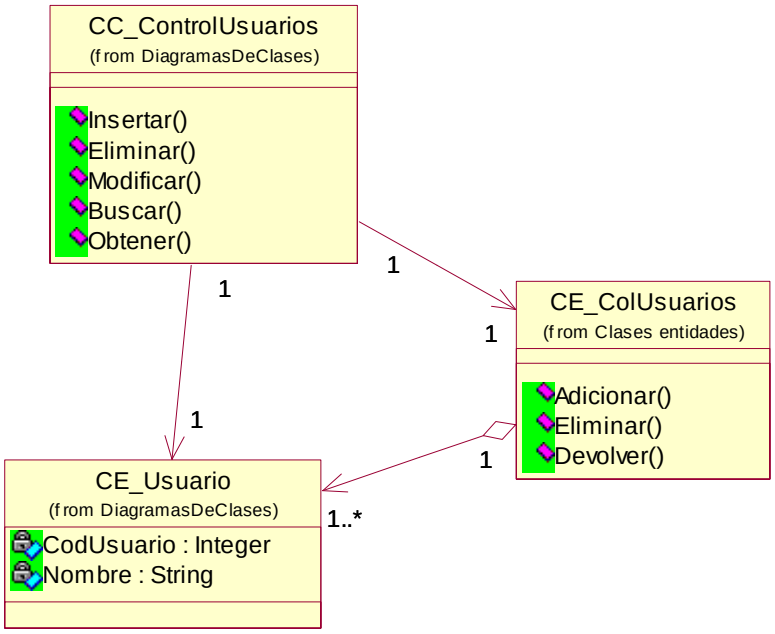


Figura 7. Diagrama de Clases del Diseño del Paquete<*Gestión de Usuarios*>

Tabla 8. Descripción de la Clase del Diseño <*CC_Control Usuarios*>

Nombre	CC_ControlUsuarios
Descripción	En esta clase se controlan todos los usuarios.
Métodos	Descripción
Insertar()	A través de este método se insertan los usuarios.
Eliminar()	A través de este método se eliminan los usuarios.
Modificar()	A través de este método se actualizan los datos de los usuarios.
Buscar()	A través de este método busca un usuario.
Obtener()	A través de este método se obtienen los usuarios que existen.

Tabla 9. Descripción de la de Clase del Diseño <*CE_ColUsuarios*>

Nombre	CE_ColUsuarios
Descripción	En esta clase se tiene el listado de usuarios.
Atributos	Tipo

Codusuario	String
Nombre	String

Tabla 10. Descripción de la de Clase del Diseño <*CE_Usuarios*>

Nombre	CE_Usuarios
Descripción	En esta clase se tiene el listado de usuarios.
Métodos	Descripción
Adicionar()	A través de este método se agregan nuevos usuarios.
Eliminar()	A través de este método se eliminan los usuarios.
Devolver()	A través de este método se obtienen los datos de los usuarios.

Diagrama de Clases del Paquete Acceso a Datos

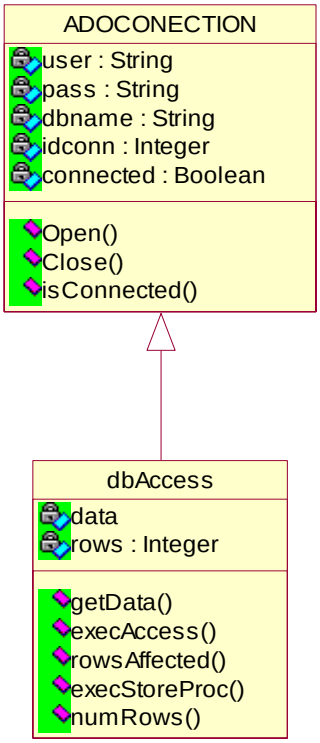


Figura 8. Diagrama de Clases del Diseño del Paquete<*Acceso a Datos*>

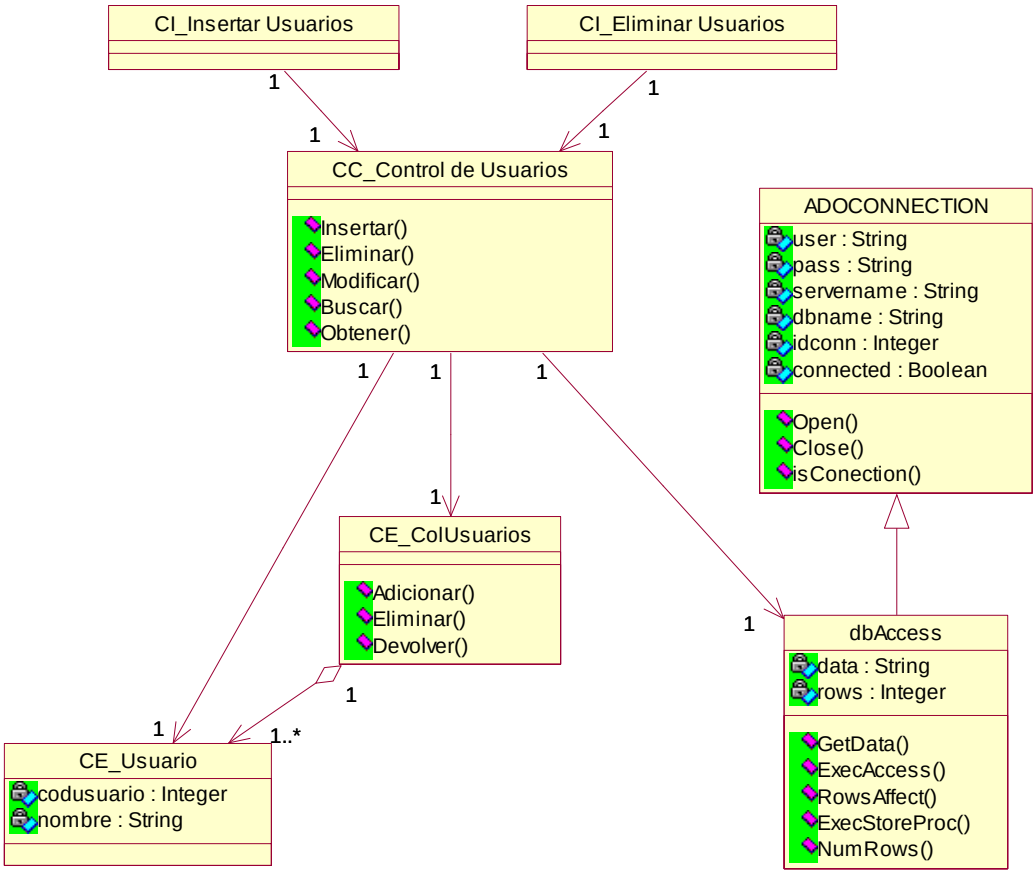


Figura 9. Diagrama de Clases del Diseño

Diagrama de Clases del Paquete Gestión de Medios

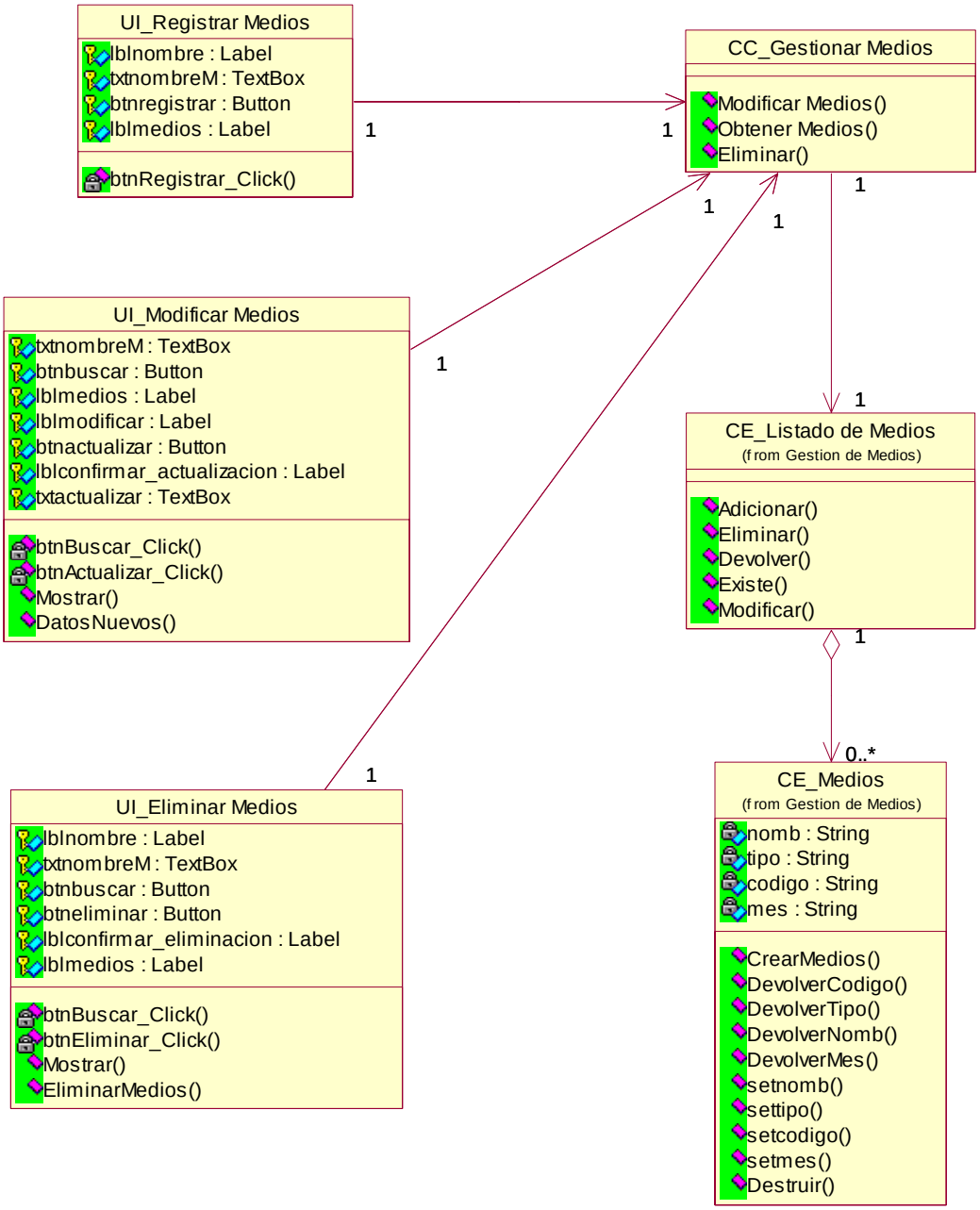


Figura 10. Diagrama de Clases para el Caso de Uso <*Actualizar Existencia de Medios*>

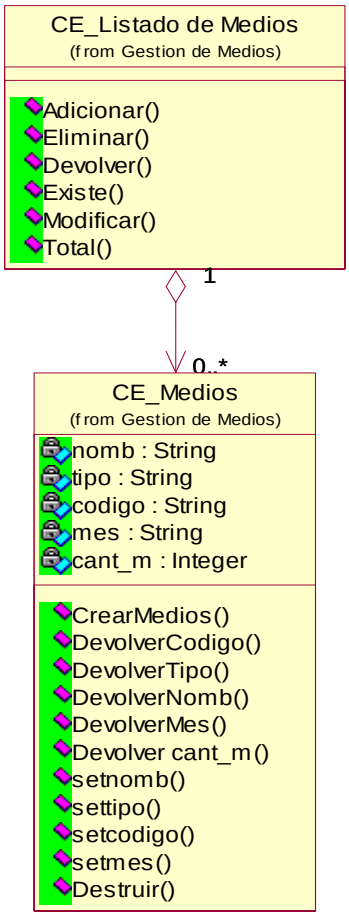


Figura 11. Aplicando Patrón Experto

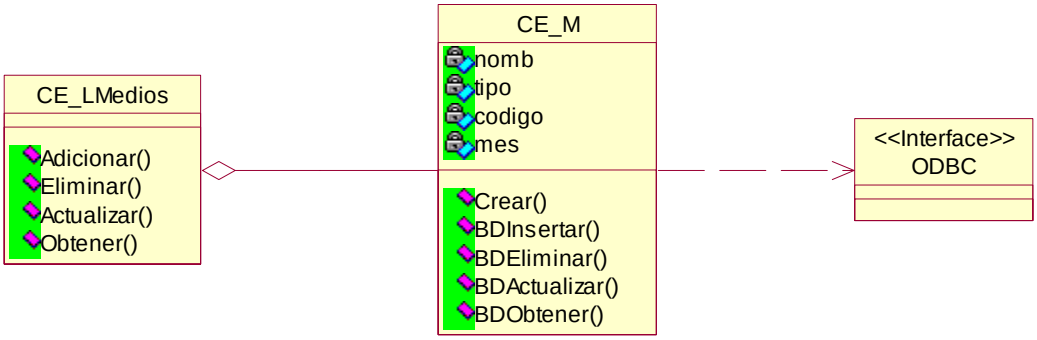


Figura 12.

Como los datos van a permanecer en la base de datos se usa la librería ODBC. La responsabilidad de registrar o recuperar la información de los medios de la BD según el patrón experto serían para la clase medios o listado de medios.

La cohesión de la clase medios ya no cumple solo con las responsabilidades del inicio sino también se encarga de registrarse en la BD, también está acoplada (depende) de la clase ODBC por tanto si se necesita reutilizar, entonces habrá que agregar la clase ODBC.

Si se aplica los patrones Alta Cohesión y Bajo Acoplamiento esta solución no es viable por tanto se utiliza el patrón "Pure Factory" o "Fabricación Pura"; con el que se crea la Clase "AP_Medios" esta clase es una fabricación de la imaginación, pero las responsabilidades asignadas a esta clase soportan alta cohesión y bajo acoplamiento. Se aprovecha el patrón de Indirección para evitar el acoplamiento entre las clases Listado de Medios y ODBC.

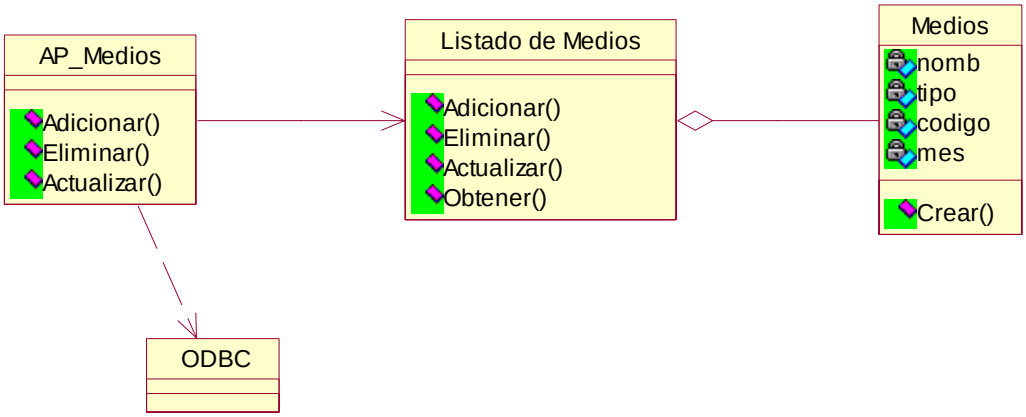


Figura 13.Aplicando Patrón "Pure Factory" e Indirección

Diagrama de Clases del Paquete Gestión de Medios

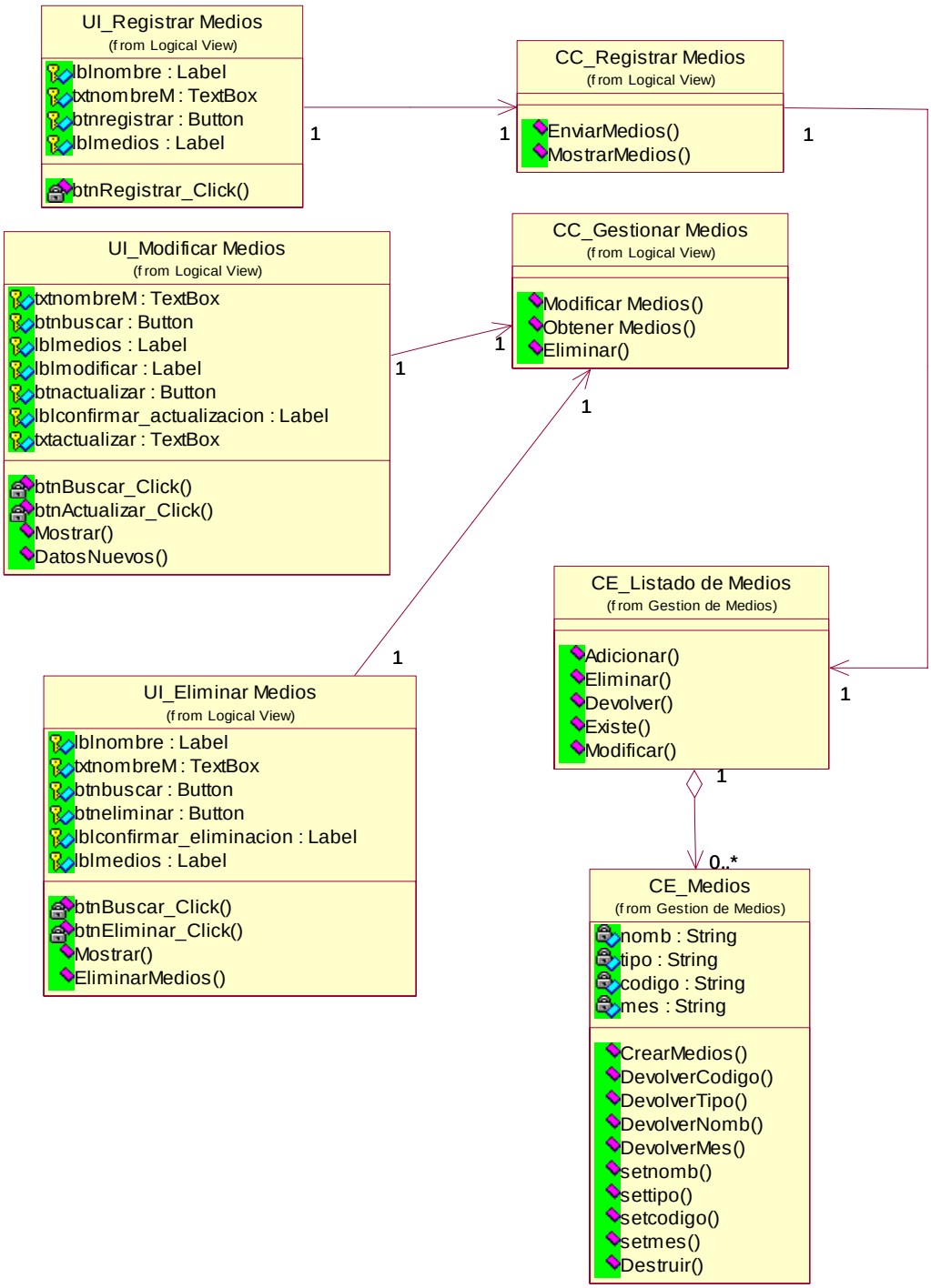
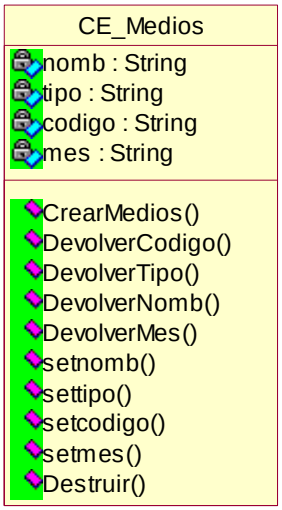


Figura 14. Diagrama de Clases del Diseño

Diseño de la Base de Datos

Modelo Lógico de Datos

1. Definir las Clases Persistentes



2. Refinar las Clases

No es obligatorio que aparezcan nuevas clases, se recomienda revisar si existe algún comportamiento de interés que no haya sido tomado en cuenta y sea trascendental en la solución del problema, para incluirlo. Se valora sobre todo las posibles relaciones de herencia -tipo es un- (generalización/especificación)

3. Clasificar las Clases y los Atributos

Tabla 10. Descripción de la Clase Persistente<CE_Medios>

Clase	CE_Medios	
Clasificación	Simple	
Atributo	Dominio	Tipo (Estático o Dinámico)
código	String	estático
nomb	String	estático
tipo	String	estático
mes	String	dinámico

4. Realizar el diagrama de clases persistentes

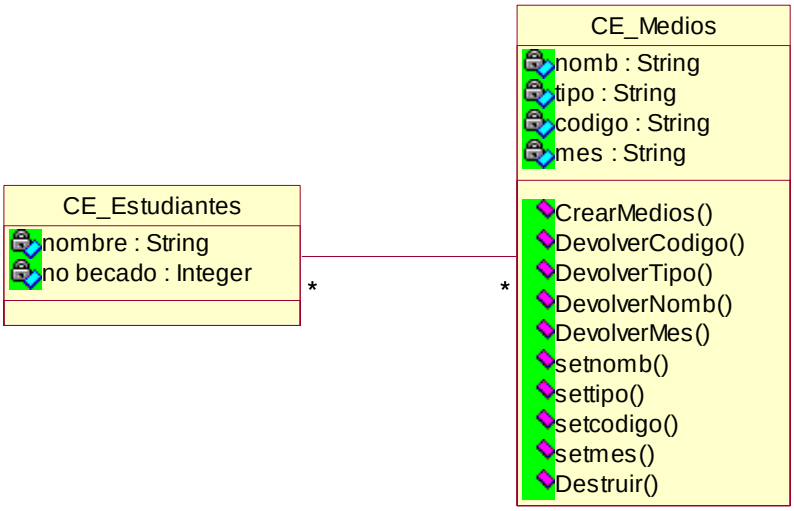


Figura 15. Diagrama de Clases Persistentes

1. Realizar el diagrama de transición de estado. (Si existe alguna clase con atributos dinámicos)
2. Convertir las clases al medio de almacenamiento.

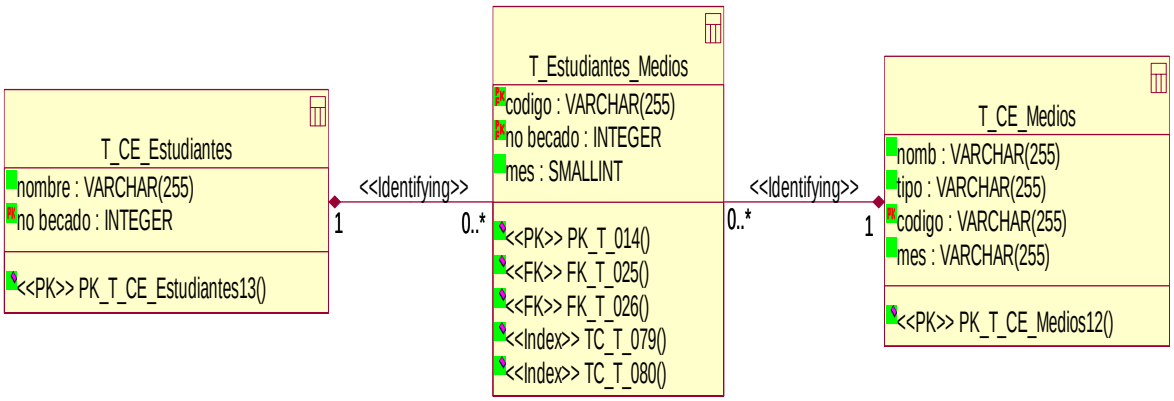
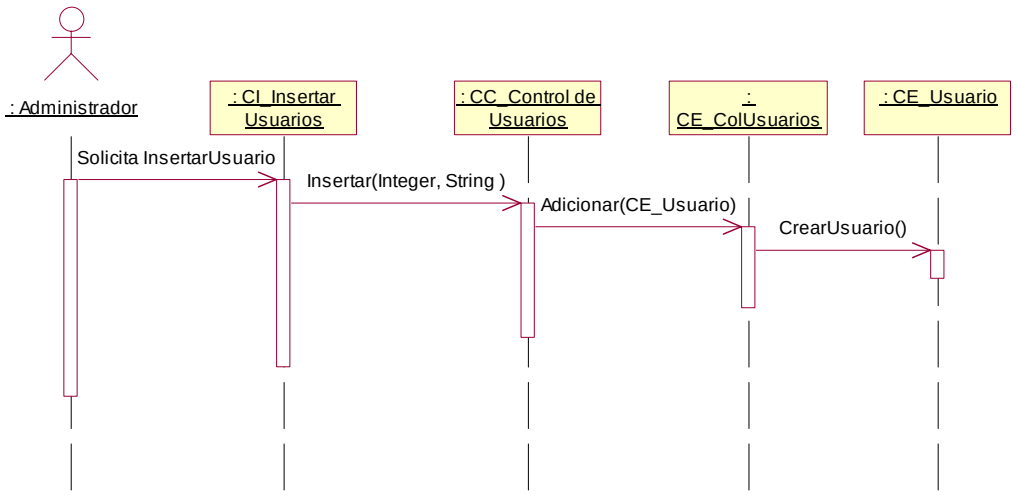


Figura 16. Modelo Lógico de Datos

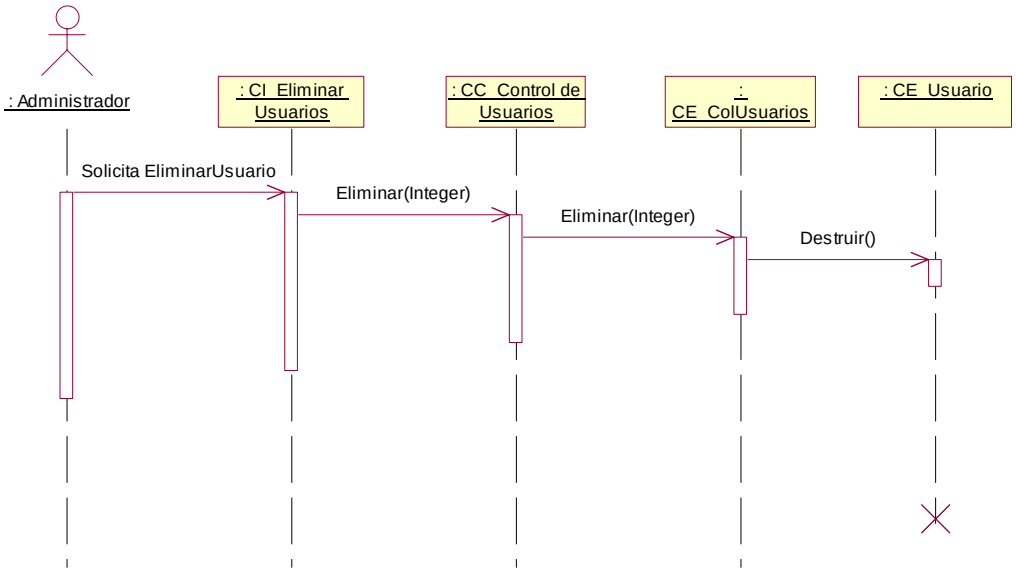
La relación de asociación se mapea a una relación de no identificación. La llave de la tabla T_CE_Listado de Medios se exporta como llave foránea a la tabla T_CE_Medios.

Diagrama de Secuencia

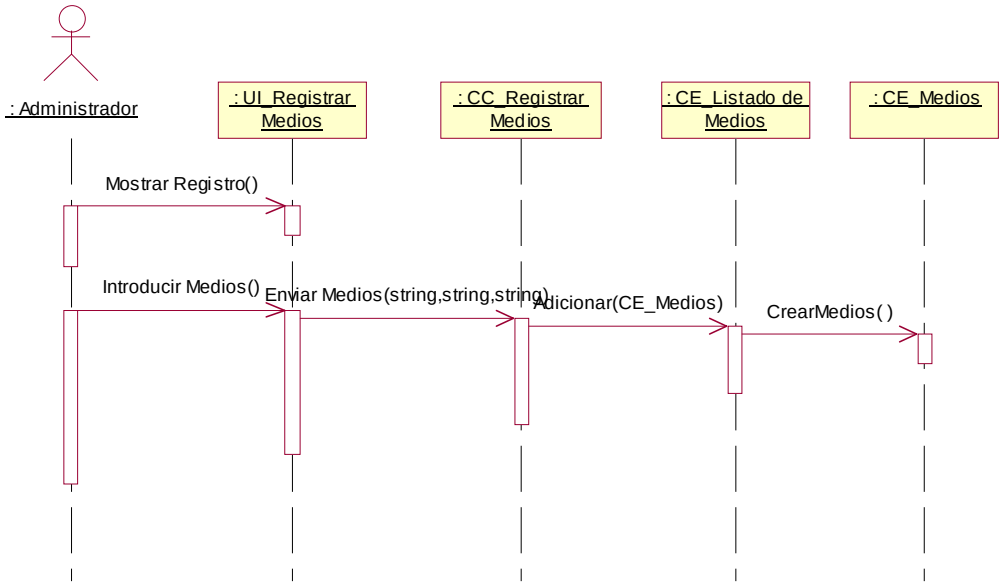
CU: Registrar usuarios. Escenario: Insertar Usuario.



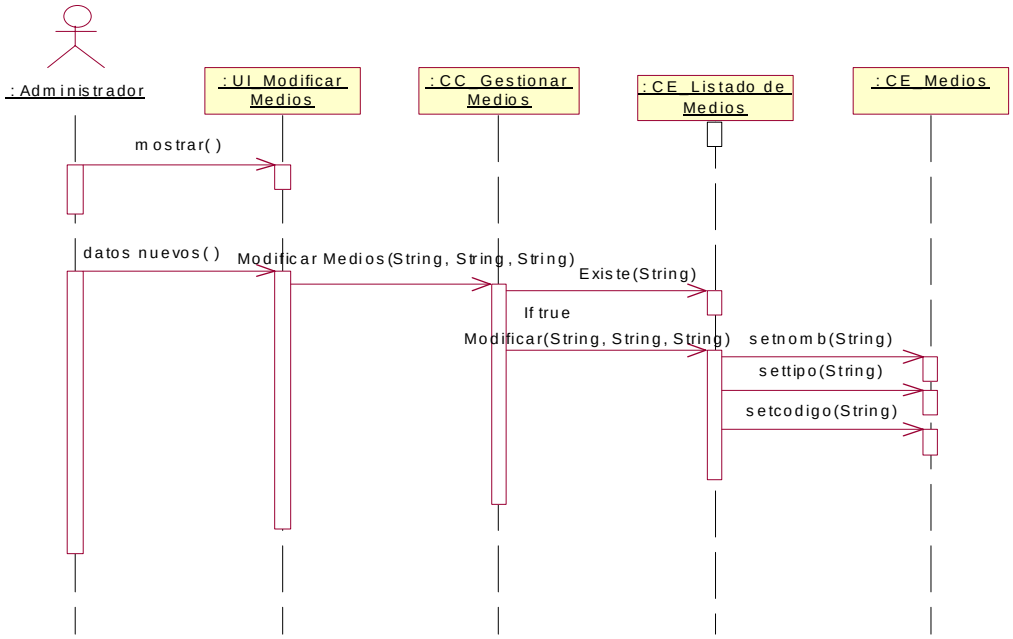
CU: Registrar usuarios. Escenario: Eliminar Usuario.



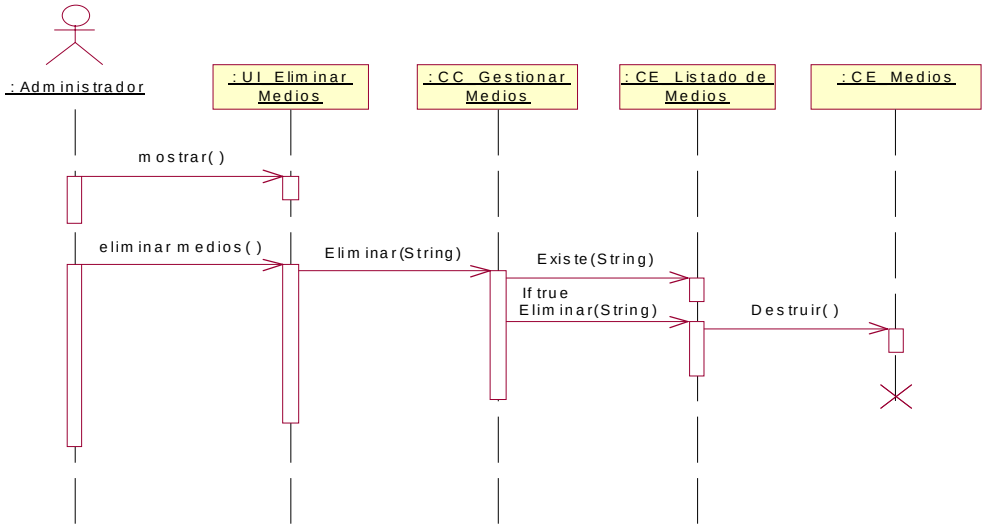
CU: Actualizar existencias de medios. Escenario: Registrar Medios.



CU: Actualizar existencias medios. Escenario: Actualizar Medios



CU: Actualizar existencias medios. Escenario: Eliminar Medios



CU: Actualizar existencias medios. Escenario: Mostrar Medios

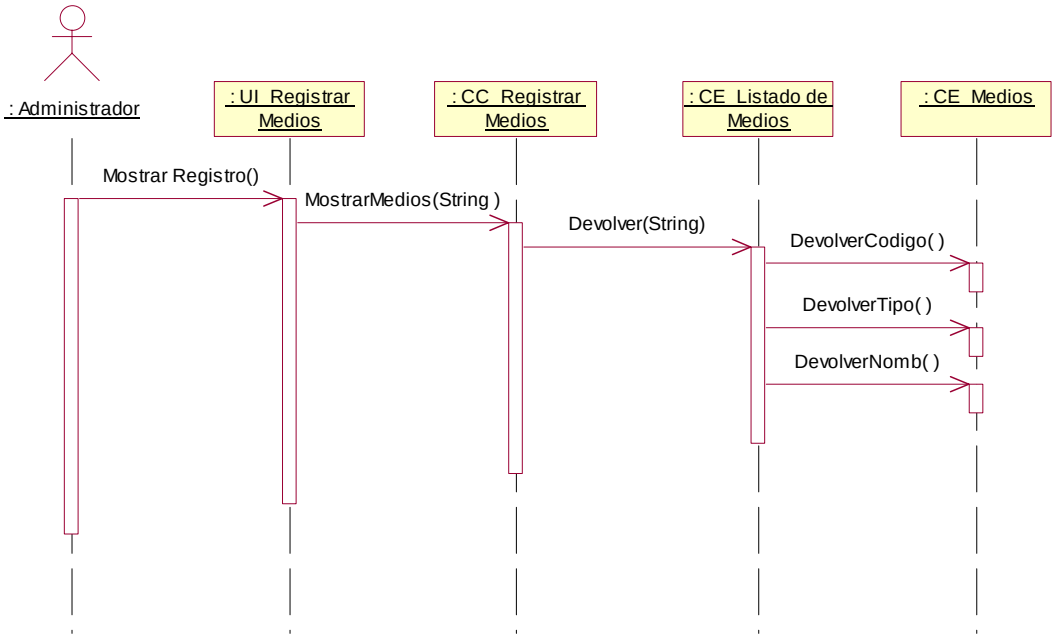


Diagrama de Despliegue

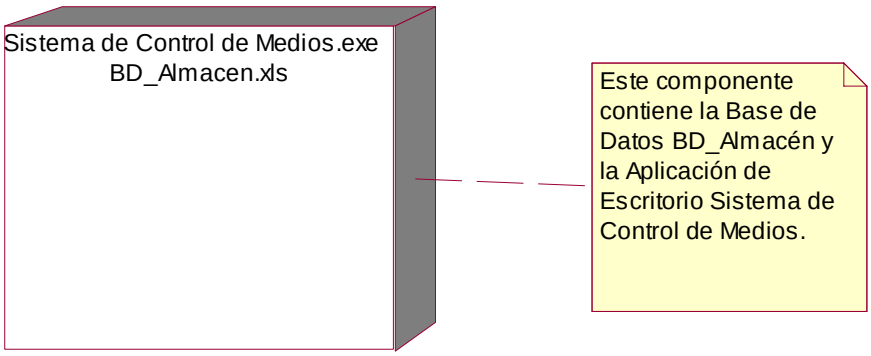


Figura 18. Diagrama de Despliegue

Diagrama de Componentes

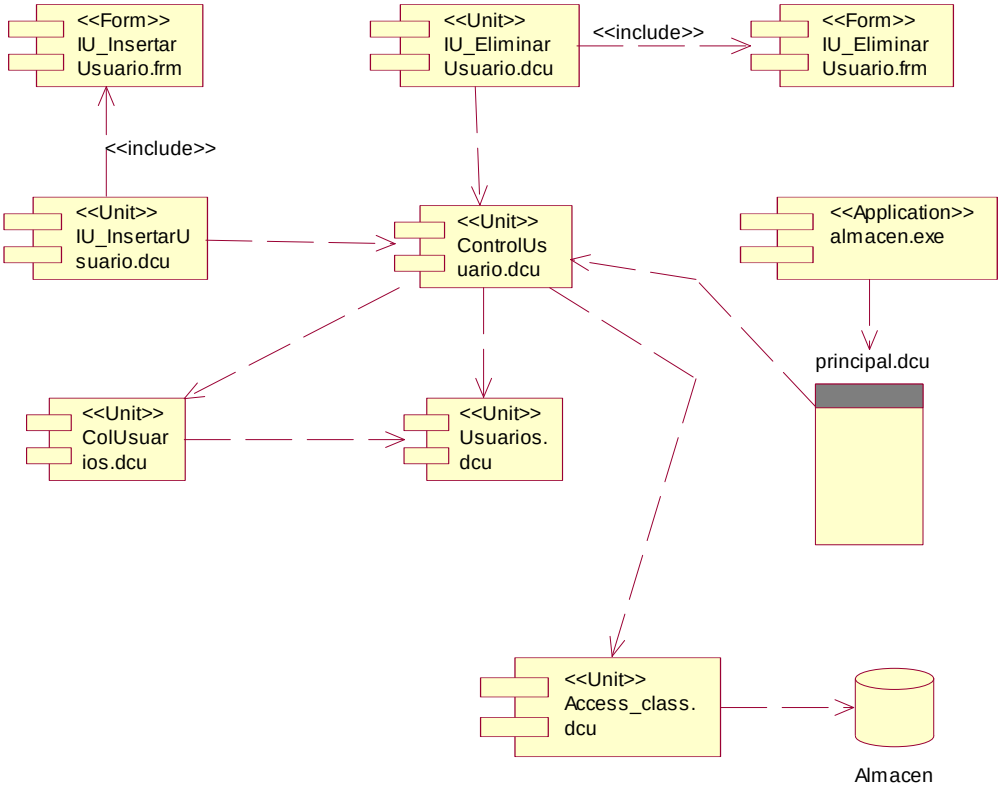


Figura 19. Diagrama de Componentes del paquete *<Gestión de Usuarios>*

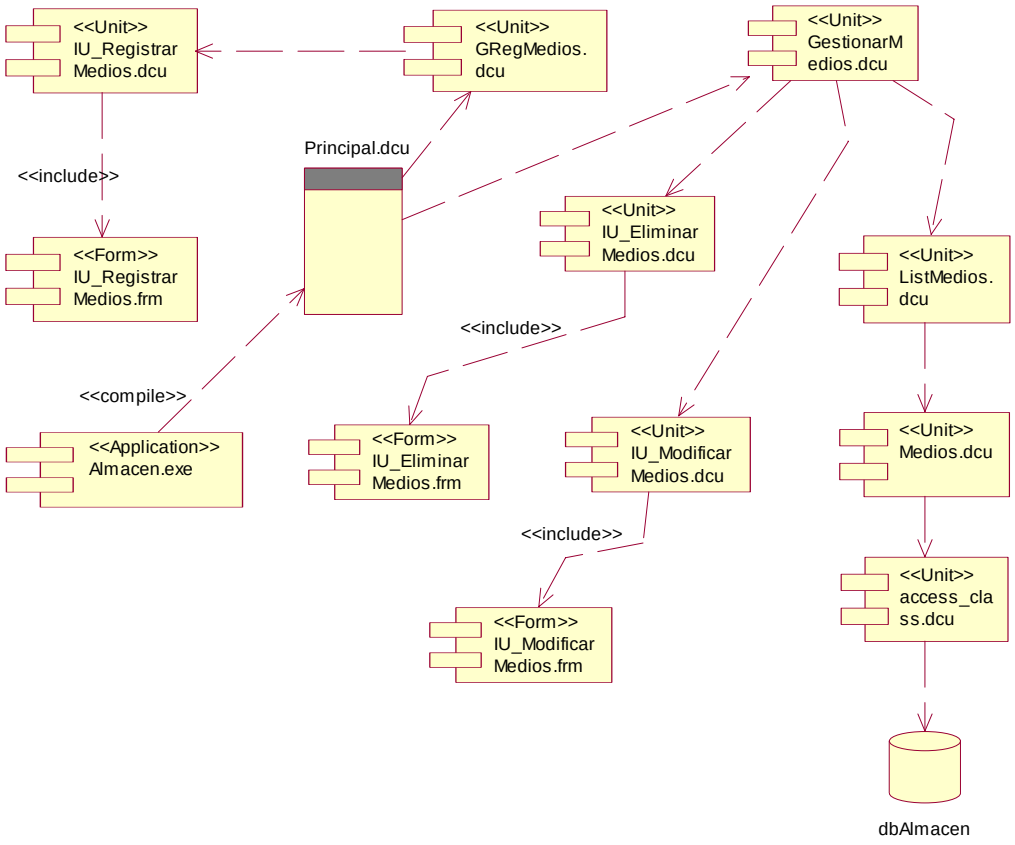


Figura 20. Diagrama de Componentes del paquete <*Gestión de Medios*>

6.3 Ejemplo para Aplicaciones Web

Este ejemplo es similar al anterior de Aplicaciones de Escritorio, pues se le realiza al mismo problema, en este se obtienen los mismos entregables, en el flujo de trabajo de Elicitación de Requisitos, los que sufren modificaciones producto al lenguaje el tipo de aplicación de muestran a continuación:

Requerimientos Funcionales y no Funcionales del Sistema

Requisitos Funcionales

La respuesta será en función solamente de la entrega de medios, entonces teniendo en cuenta los objetivos de los futuros usuarios del sistema y la descripción de cómo debe funcionar el mismo, se pueden inferir los requerimientos funcionales siguientes:

El sistema debe ser capaz de:

R 1-Autentificar usuario: Permitir a los usuarios acceder a la información que le corresponde.

R1.1- Comparar Usuario y contraseña con los usuarios del sistema.

R1.2- Asignar privilegios.

R 2- Controlar Existencia de medios.

R2.1- Registrar los tipos de medios adquiridos.

R 2.2- Actualizar datos de los medios adquiridos.

R 2.3- Eliminar medios.

R 2.4- Listar medios existentes en el almacén.

R 3- Controlar tenencia de medios.

R 3.1- Buscar al estudiante por su carnet de becado.

R 3.2- Comprobar medios pendientes.

R 3.3- Mostrar Lista de medios disponibles.

R 3.4- Registrar la entrega.

R 4-Visualizar Medios

Requisitos no Funcionales

1 Apariencia o interfaz externa

La herramienta propuesta será usada por personas que no necesariamente tienen habilidades en el trabajo en la computadora, por lo que la interfaz debe ser amigable y fácil de usar, de manera que no sea una dificultad para el usuario el uso de ella. Cada página no debe exceder 300Kb.

2 Usabilidad

El sistema debe ser concebido para dar facilidad de uso a personas sin experiencia previa.

3 Rendimiento

Para un funcionamiento óptimo de la aplicación se seguirán las diferentes técnicas de elaboración en la Web, que faciliten el rápido acceso a sus páginas. La eficiencia del producto estará determinada en gran medida por el aprovechamiento de los recursos que se disponen en el modelo Cliente/Servidor, y la velocidad de las consultas en la Base de Datos.

La herramienta propuesta debe ser rápida y el tiempo de respuesta debe ser el mínimo posible, adecuado a la rapidez con que el cliente requiere la respuesta a su acción.

5 Portabilidad

La herramienta propuesta podrá ser usada bajo cualquier sistema operativo, para su implementación se usaron Herramientas de Programación y Gestión de Bases de Datos que son multiplataforma.

6 Seguridad

El sistema debe comunicarse usando un protocolo seguro (https). Los datos no pueden viajar de forma transparente por la red.

7 Políticos-culturales

La herramienta propuesta deberá responder a los intereses de la Constitución de la República de Cuba, asimismo no existirán prioridades en el servicio según el nivel social, cultural o étnico.

8 Legales

Cómo el sistema fue creado en un lenguaje multiplataforma no tendrá problemas ya que corre en cualquier sistema operativo.

9 Confiabilidad

La información manejada por el sistema está protegida de acceso no autorizado y divulgación.

10 Software

En la computadora que haga función de servidor, independientemente del sistema operativo, se necesita el lenguaje de programación PHP y el SGBD, MySQL. En las computadoras de los usuarios y del grupo de soporte sólo se requiere de navegador para Internet o Intranet. Se utilizará un servidor Web Apache.

13 Hardware

Se requiere de un servidor de 128 MB de RAM como mínimo y # GB de capacidad del disco duro, todas las computadoras implicadas, tanto para la administración como las de los usuarios, deben estar conectados a una red y tener al menos 64 MB de RAM.

14 Restricciones en el Diseño e Implementación

Para ser consecuente con el planteamiento de hacer una herramienta que pueda ser usada por cualquier usuario es necesario usar para su implementación lenguajes de programación que sean multiplataforma, en este caso el PHP. Por esta misma razón es necesario usar un Sistema Gestor de Bases de Datos multiplataforma, en este caso MySQL.

Para garantizar una mejor documentación del sistema, así como el uso de última tecnología, se utiliza para realizar el análisis y el diseño del sistema UML (Unified Modelling Language) y su extensión para el desarrollo de proyectos Web. Como herramienta de apoyo a este Lenguaje de Modelación se utiliza Rational Rose.

Análisis y Diseño

También presenta los mismos artefactos que la Aplicación de Escritorio en el flujo de Análisis y Diseño, solo cambian algunos que se muestran a continuación.

Tabla 4. Definición de Actores del Sistema a automatizar

Nombre del actor	Descripción
Estudiante	Persona que va a utilizar el sistema para buscar información sobre los medios.
Distribuidora	Es la encargada de llevar a cabo todo lo relacionado con la entrega de Medios a través de la aplicación. Actualiza la ficha de medios de cada estudiante en el sistema.
Administrador	Responsable de realizar todas las actualizaciones de existencia en el almacén; registra, elimina y modifica datos acerca de los medios en el sistema. Permite la autenticación de los usuarios al sistema.

Diagrama de Casos de Uso del Sistema

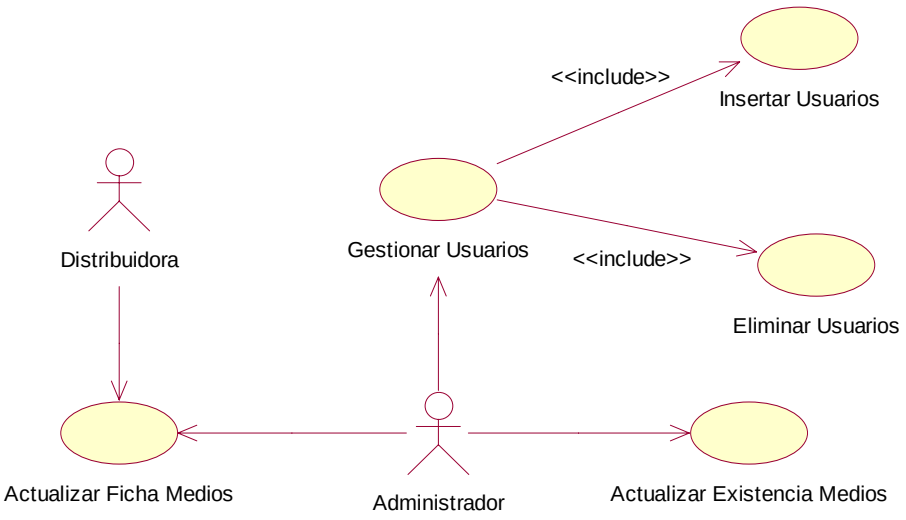


Figura 21. Diagrama del Caso de Uso del Sistema

Descripción de los Casos de Uso del Sistema

Descripción de los Casos de Uso del Sistema que se agregan:

Tabla 5. Descripción del Caso de Uso del Sistema <Autenticar usuarios>

Nombre del Caso de Uso	Autenticar usuarios	
Actores	Administrador, Distribuidora, Estudiante	
Propósito	Permitir a los usuarios acceder a la información que le corresponde.	
Resumen	El Caso de Uso se inicia cuando el administrador introduce los datos que se le piden del usuario para que este pueda acceder a la aplicación, estos se verifican y finaliza dándole los permisos y autorizándole la entrada.	
Referencias	R1	
Precondiciones		
Poscondiciones	Se autorizan las funcionalidades según los privilegios.	
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El administrador entra Usuario y Contraseña.		1.1 El sistema encripta la contraseña. Busca el usuario y compara la contraseña. 1.2 En caso de ser correcto se le asignan los permisos.
Curso alternativo		
Acciones del Actor		Respuesta del Sistema
		1.3 En caso de no existir se envía un mensaje de aviso.
Prioridad	Crítico	

Tabla 8. Descripción del Caso de Uso del Sistema <Visualizar Medios>

Nombre del Caso de Uso	Visualizar Medios	
Actores	Estudiante	
Propósito	Mostrar los medios que le corresponden.	
Resumen	El caso de uso se inicia con la presentación de los medios asignados al estudiante.	
Referencias	R4	
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El estudiante del sistema solicita ver sus medios.		1.1 El sistema carga la presentación de los medios.
Curso alternativo		
Prioridad	Secundario	

Diagrama de Clases del Paquete Acceso a Datos

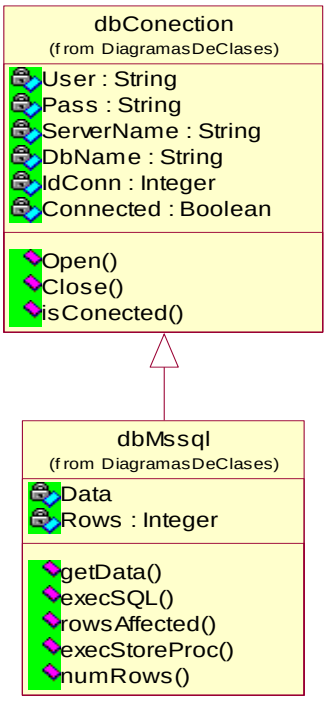


Figura 22. Diagrama de Clases del Paquete <Acceso a Datos>

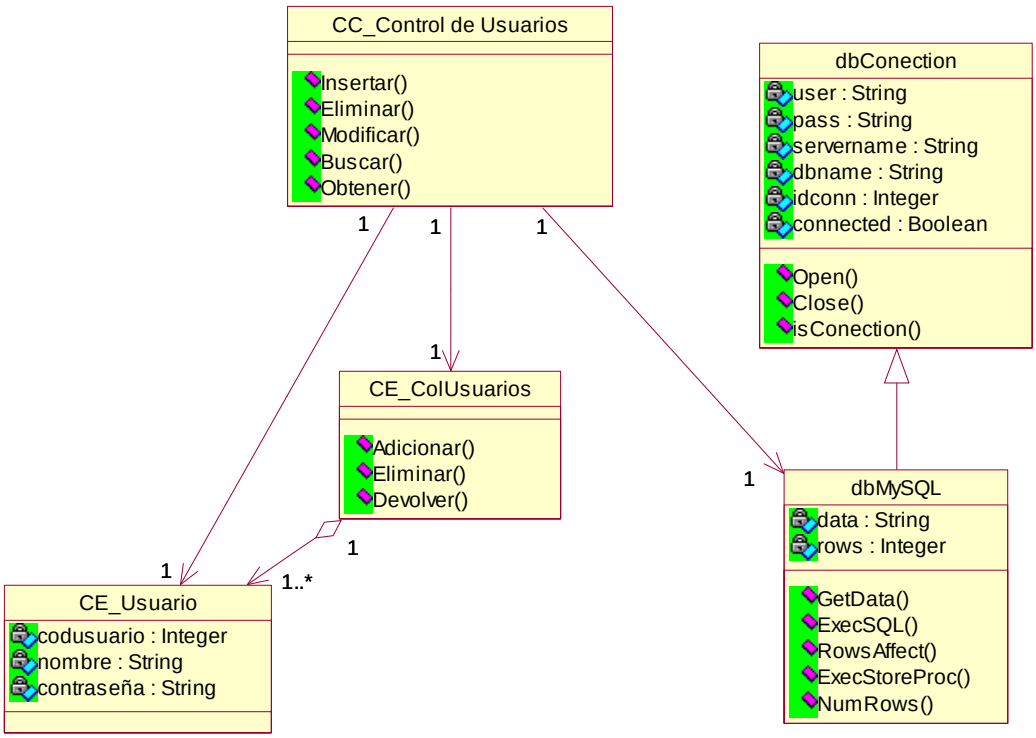


Figura 23. Diagrama de Clases del Diseño

Diagrama de Clases del Paquete Gestión de Estudiantes

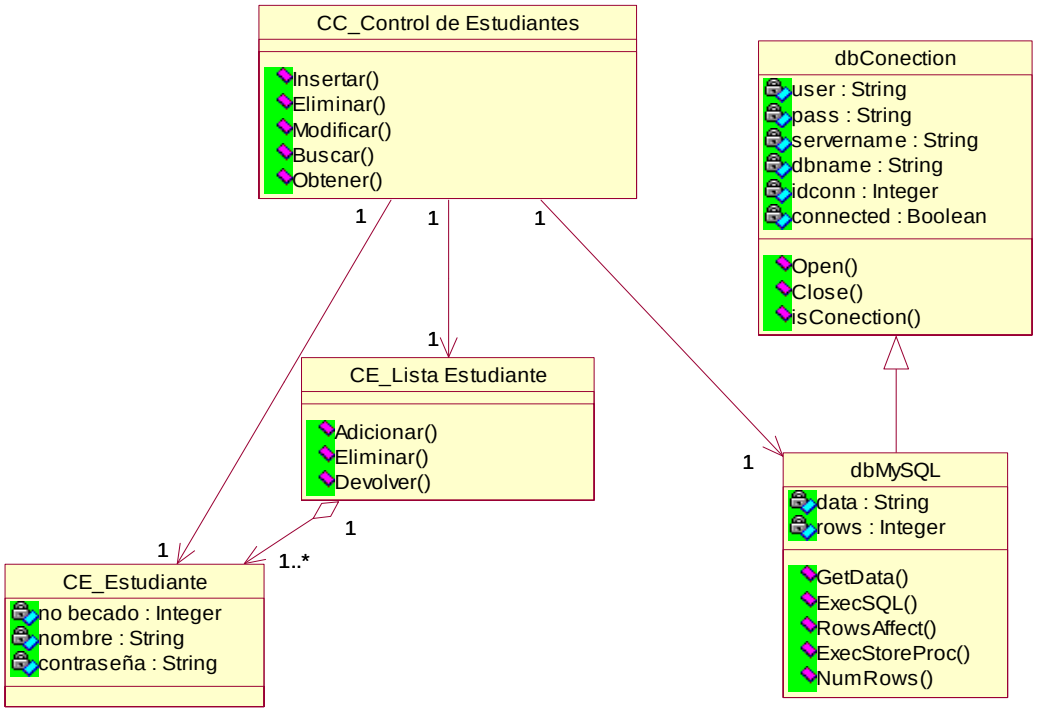


Figura 24. Diagrama de Clases del Diseño

Diagrama de Clases del Paquete Gestión de Medios

De este paquete los diagramas que se modifican son:

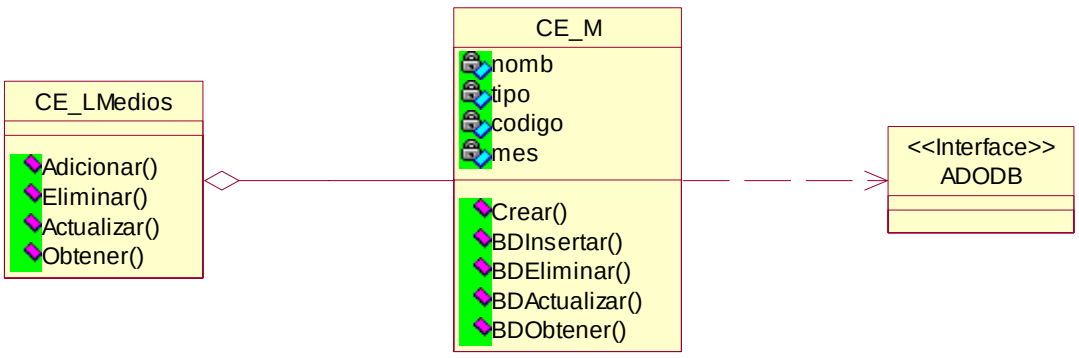


Figura 25. Aplicando Patrón Experto

Como los datos van a permanecer en un gestor de base de datos relacional se usa la librería ADODB. La responsabilidad de registrar o recuperar la información de los medios de la BD según el patrón experto serían para la clase medios o listado de medios.

La cohesión de la clase medios ya no cumple solo con las responsabilidades del inicio sino también se encarga de registrarse en la BD, también está acoplada (depende) de la clase ADODB por tanto si se necesita reutilizar, entonces habrá que agregar la clase ADODB.

Si se aplica los patrones Alta Cohesión y Bajo Acoplamiento esta solución no es viable por tanto se utiliza el patrón "Pure Factory" o "Fabricación Pura"; con el que se crea la Clase "AP_Medios" esta clase es una fabricación de la imaginación, pero las responsabilidades asignadas a esta clase soportan alta cohesión y bajo acoplamiento. Se aprovecha el patrón de Indirección para evitar el acoplamiento entre las clases Listado de Medios y ADODB.

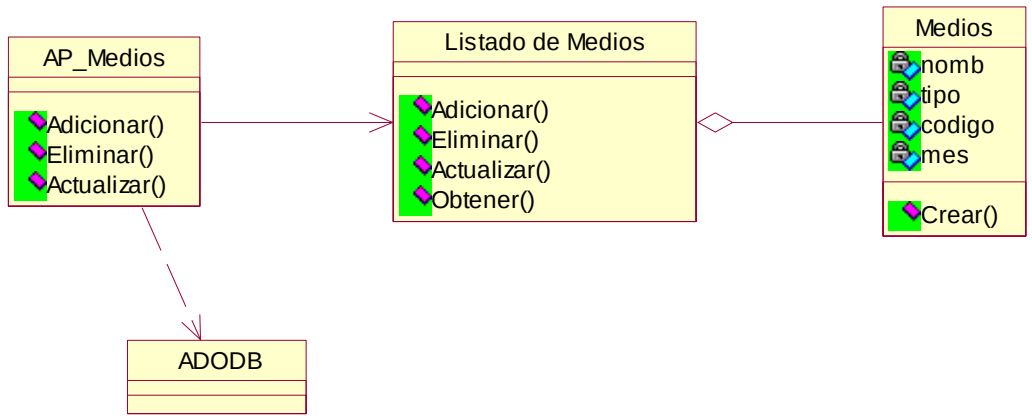


Figura 26. Aplicando Patrón "Pure Factory" e Indirección

Diagrama de Clases Persistentes

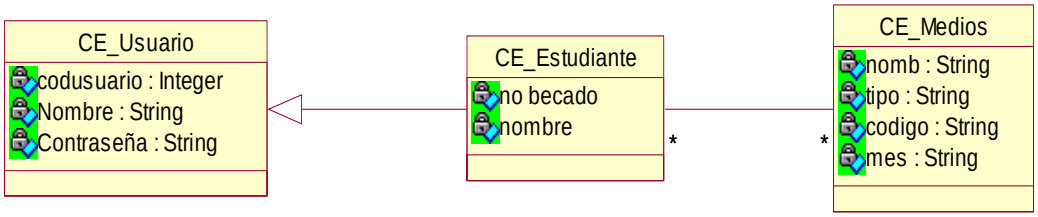


Figura 27. Diagrama de Clases Persistentes

Modelo de Datos

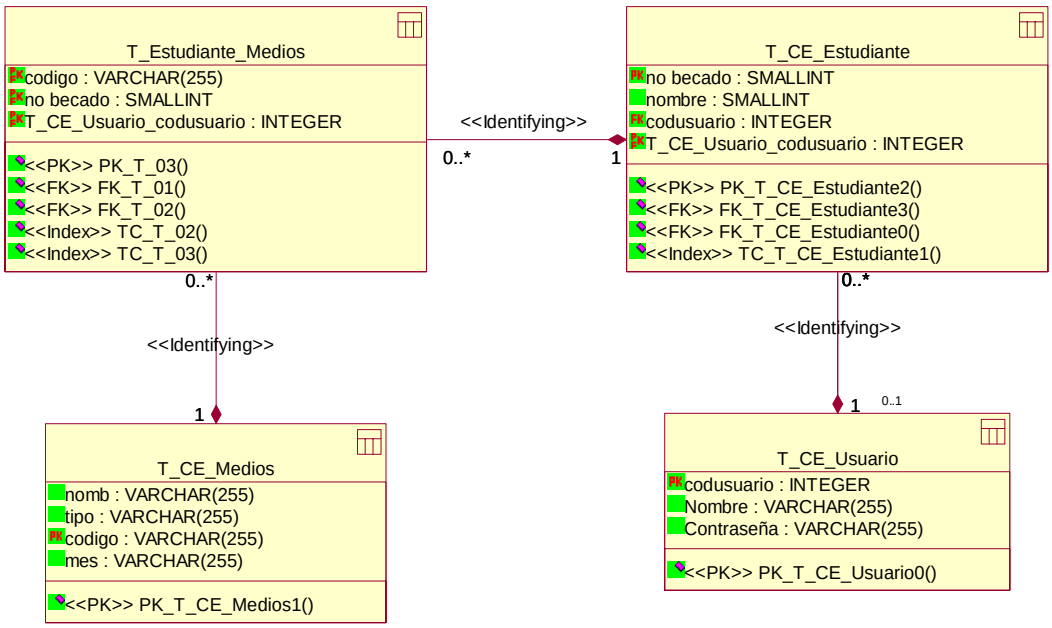


Figura 28. Diagrama de Clases Persistentes

Debido a que es una Aplicación Web se incrementan los artefactos siguientes: Diagrama de Clases Web y Mapa de Navegación.

Diagrama de Clases Web

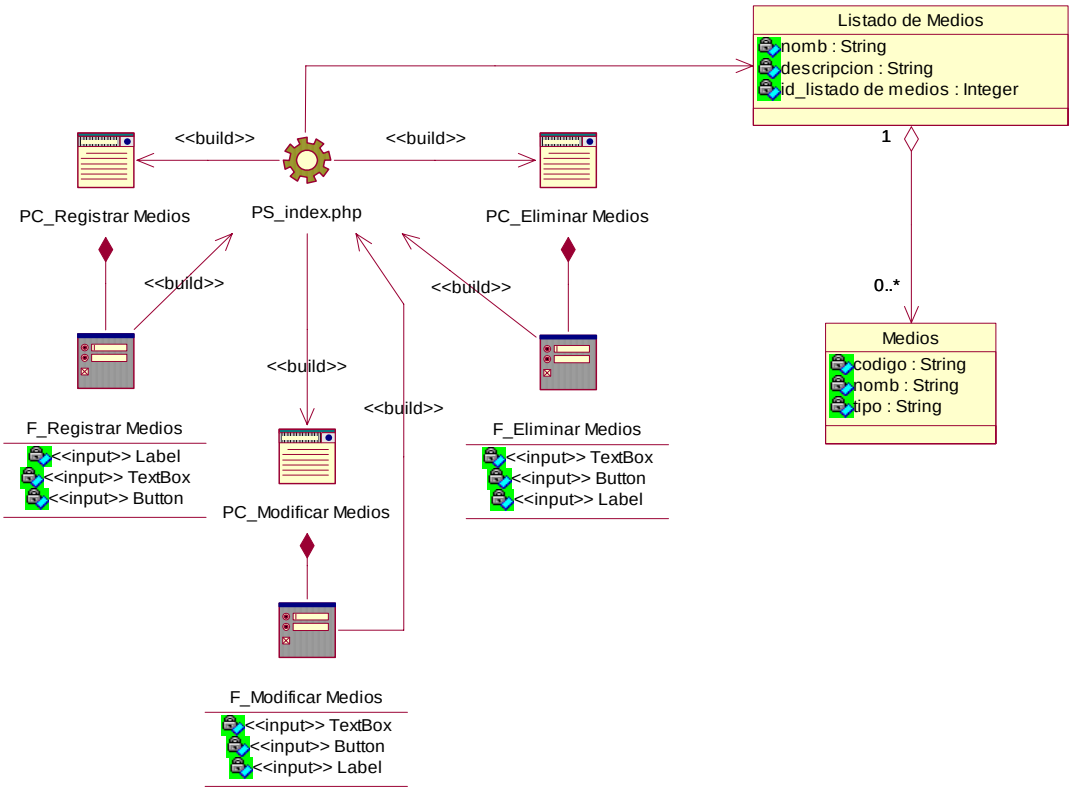


Figura 29. Diagrama de Clases Web

Mapa de Navegación

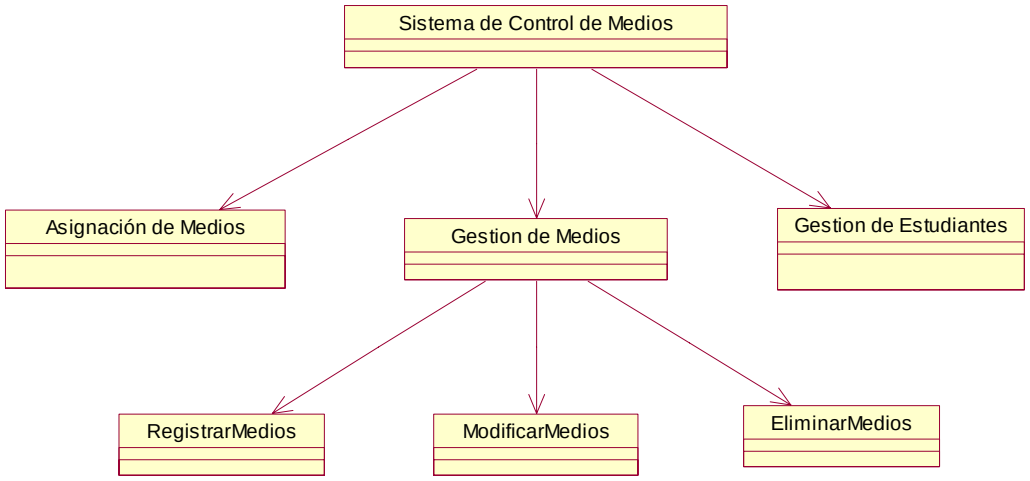


Figura 28. Mapa de Navegación

Diagrama de Presentación

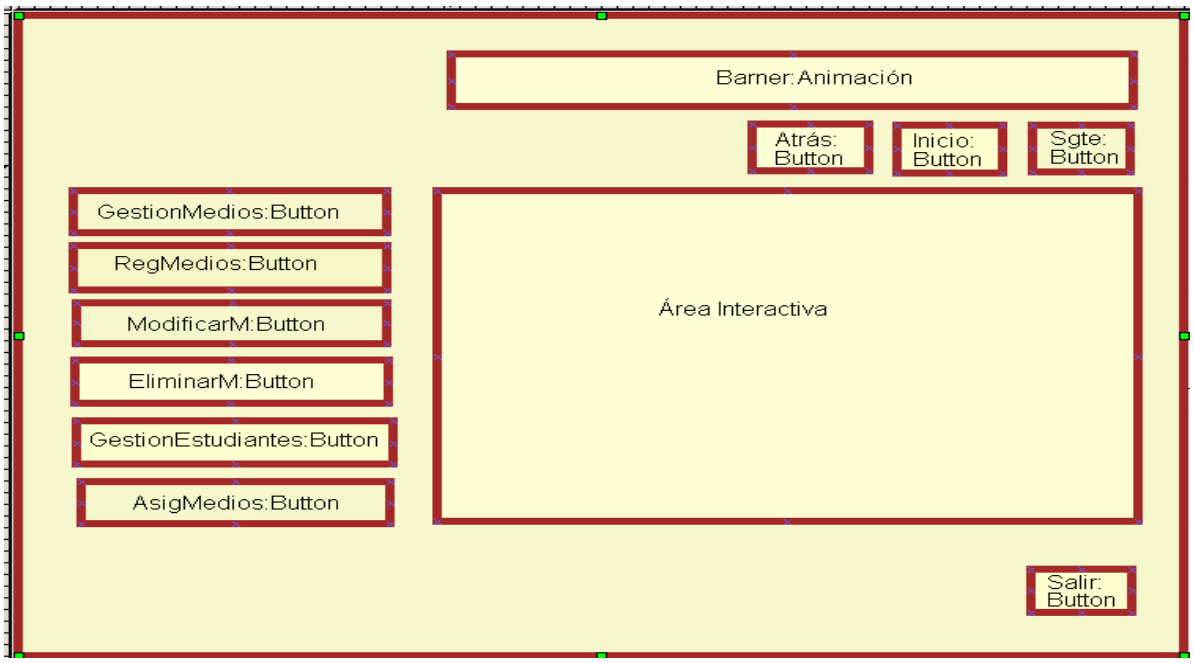


Figura 29. Diagrama de Presentación

Implementación

Diagrama de Despliegue

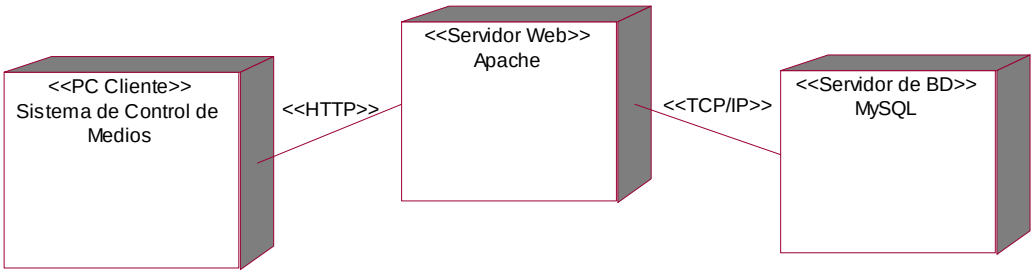


Figura 30. Diagrama de Despliegue

Diagrama de Componentes

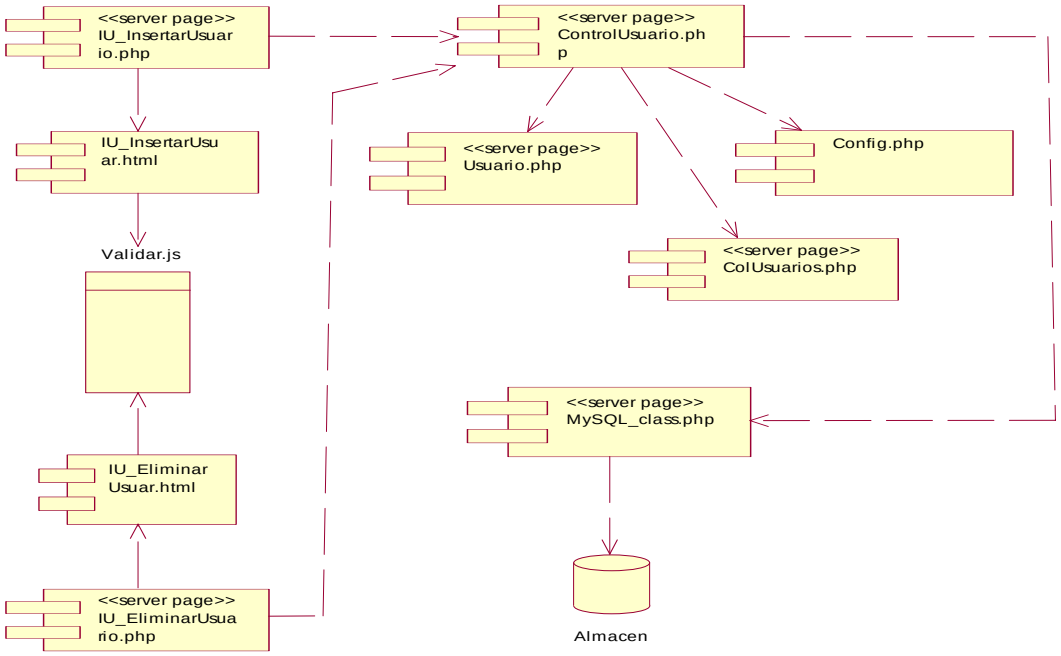


Figura 31. Diagrama de Componentes del paquete <*Gestión de Usuarios*>

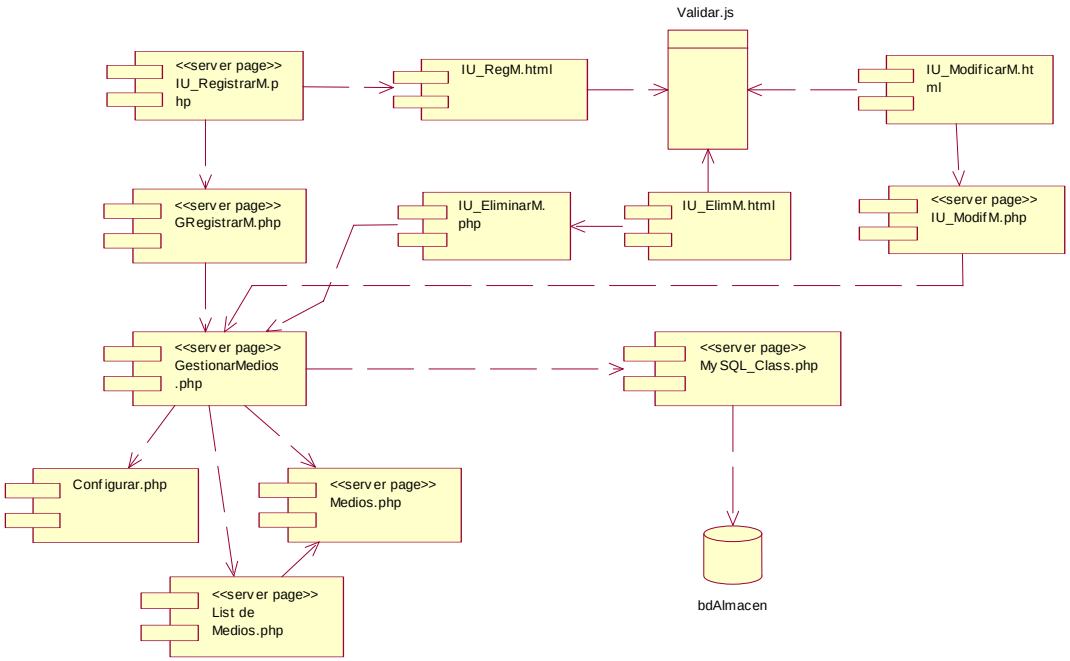


Figura 32. Diagrama de Componentes del paquete <*Gestión de Medios*>

6.4 Ejemplo para Multimedia

Descripción del Modelo de Dominio

Debido a la poca estructuración de los procesos de negocio se plantea un modelo de dominio ayudando a una mejor compresión de los conceptos del sistema. Para esto se realiza la descripción del modelo del dominio a través de un diagrama de clases UML, en el cual se definen las principales clases conceptuales que intervienen en el sistema.

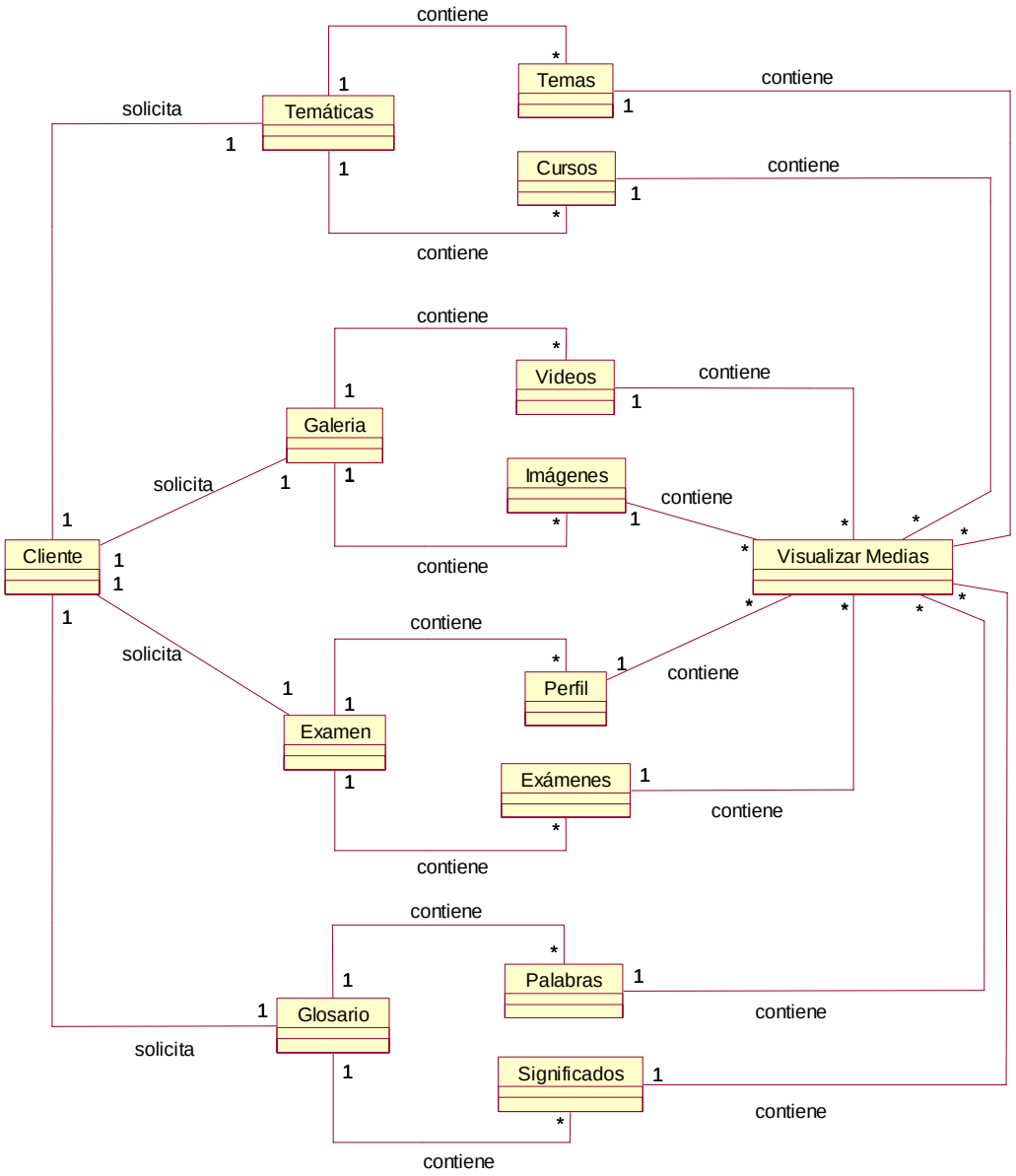


Figura 30. Modelo de Dominio

Identificación de conceptos que se utilizarán en el diagrama:

• Se le denominará cliente a cualquier usuario que interactúe con el sistema.

• Se le denominará temáticas al objeto que contiene los tópicos: temas y cursos relacionados con el idioma.

• Se le denominará glosario al objeto que contiene las palabras y significados fundamentales del idioma Inglés.

• Se le denominará galería al objeto que contiene las medias, tanto videos como imágenes.

• Se le denominará examen al objeto que recoge los datos del cliente y posibilita seleccionar un examen.

• Se le denominará visualizar media al objeto que se encarga de visualizar todas las medias ya sean videos, imágenes o audio.

Solución propuesta

Elaboración de un sistema multimedia que conste con cuatro paquetes (Presentación, Temáticas, Glosario y Examen), los cuales centran la información referida al Inglés como idioma fundamental a dominar.

Requisitos Funcionales

Presentación

R1_ Mostrar presentación particular de la aplicación.

Temáticas

R2_ Mostrar el contenido que se aborda en “temas”.

R3_ Mostrar el contenido que se aborda en “cursos”.

Glosario

R4_ Mostrar listado de palabras fundamentales que se utilizan para dominar el Inglés.

R5_ Mostrar significado de la palabra seleccionada.

Galería

R6_ Mostrar los contenidos referidos al idioma de Inglés en formato de video.

R7_ Mostrar los contenidos referidos al idioma de Inglés en imágenes.

Examen

R8_Obtener los datos del cliente para mostrar una información personalizada.

R9_ Autorizar la selección del test que se va a realizar.

R10_ Presentar los resultados alcanzados en la realización de los test.

Requisitos Generales

R11_ Permitir el control de audio del sistema.

R12_ Permitir el retorno a la pantalla principal.

R13_ Permitir la manipulación de la información mostrada en videos y audio.

R14_ Mostrar el contenido ofrecido en la ayuda cuando sea solicitada.

R15_ Permitir en los controladores de medias las opciones de: ejecutar, pausar y detener.

R16_ Permitir el acceso a los módulos comprendidos en el sistema.

R17_ Permitir la salida del sistema cuando sea solicitada.

Requisitos no Funcionales

Resolución de pantalla, profundidad de colores

El producto deberá imponer los requerimientos de resolución y profundidad de colores:

• La resolución de pantalla es de 800 x 600 pixels.

• La profundidad de color será de 24 bits.

Navegación

• De cualquier pantalla se podrá acceder a cualquier otro módulo de la aplicación.

• De cualquier pantalla se podrá salir o abandonar la aplicación, una vez que el cliente lo haya confirmado previamente.

Prestaciones generales

• Las prestaciones generales como: audio, ayuda, salir, etc, estarán disponibles para el cliente siempre y cuando la navegación se efectué por las pantallas del sistema.

Sistema Operativo

• Para la ejecución de la aplicación se necesita que el sistema sea multiplataforma.

Modelo de Casos de Uso del Sistema

Actores del Sistema

Tabla 11. Descripción de los Actores del Sistema

Nombre del actor	Descripción
Cliente	Representa a una persona que va a utilizar el sistema para buscar información sobre alguna temática determinada.

Descripción y expansión de los casos de uso

Presentación

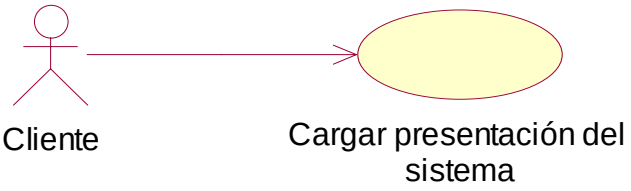


Figura 31. Diagrama de Caso de Uso de Presentación

Referencia	Caso de Uso	Prioridad
CUS_ 1	Cargar presentación del sistema	Secundario

Tabla 12. Descripción del caso de uso del sistema <*Cargar presentación del sistema*>

CUS_1	Cargar presentación del sistema
Actores	Cliente
Propósito	Mostrar la presentación de la aplicación.
Resumen	Se inicia con la presentación general de la aplicación, la cual será de obligatoria visualización por parte del cliente. El cursor del ratón en esta, no estará visible y ninguna acción por parte del usuario podrá interrumpir la misma. Al culminar la presentación de la aplicación se dará paso automáticamente a la pantalla principal del producto.
Referencias	R1
Precondiciones	
Poscondiciones	Esta presentación se mostrará una sola vez, ya que es la

	inicialización de la aplicación.
Curso Normal de los Eventos	
Acciones del Actor	Respuesta del Sistema
1. El cliente del sistema solicita comenzar a trabajar en la multimedia.	1.1 El sistema carga la presentación de la Multimedia Inglés Fácil.
Curso alternativo	

Generales

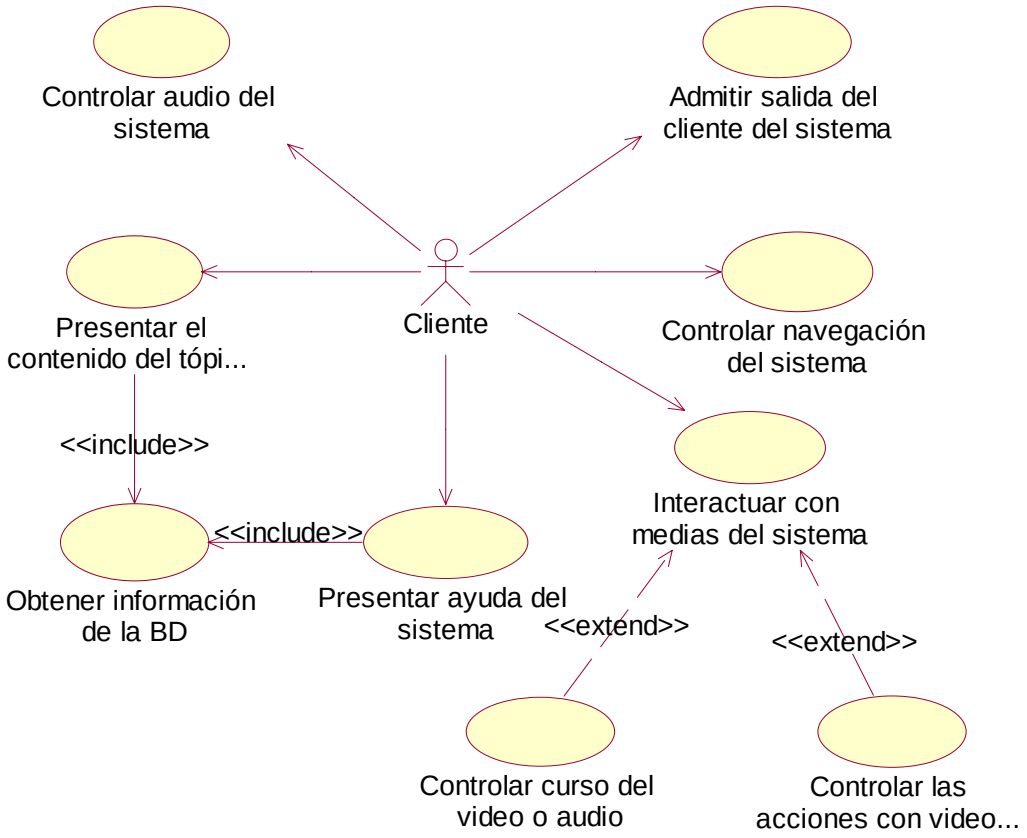


Figura 32. Diagrama de Caso de Uso de Generales

Referencia	Caso de Uso	Prioridad
CUS_ 2	Controlar audio del sistema	Secundario
CUS_ 3	Presentar contenido del tópico seleccionado	Crítico
CUS_ 4	Controlar navegación del sistema	Crítico

CUS_ 5	Presentar ayuda del sistema	Crítico
CUS_ 6	Admitir salida del cliente del sistema	Secundario
CUS_ 7	Interactuar con medias del sistema	Crítico
CUS_ 8	Controlar curso de video o audio	Secundario
CUS_ 9	Controlar las operaciones con video o audio	Crítico
CUS_ 10	Obtener información de la BD	Secundario

Tabla 13. Descripción del caso de uso del sistema <*Controlar audio del sistema*>

CUS_2	Controlar audio del sistema	
Actores	Cliente	
Propósito	Permitir la manipulación del audio.	
Resumen	El caso de uso se inicia cuando el cliente solicita la opción de control de audio del sistema.	
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente estando en cualquier pantalla, solicita manipular el audio.		1.1. El sistema se encarga de realizar la manipulación correspondiente.
Curso alternativo		

Tabla 14. Descripción del caso de uso del sistema <*Presentar contenido del tópico seleccionado*>

CUS_3	Mostrar contenido del tópico seleccionado.
Actores	Cliente
Propósito	Mostrar la información referida al tópico seleccionado.
Resumen	El caso de uso se inicia cuando el cliente solicita información acerca de los tópicos centrales que ofrece la multimedia, luego el sistema se encarga de obtener y mostrar la información solicitada.
Precondiciones	Que haya culminado el caso de uso Mostrar presentación del sistema.
Poscondiciones	El cliente solo podrá interactuar con una pantalla del

	tópico, el que corresponda a la opción seleccionada.
Curso Normal de los Eventos	
Acciones del Actor	Respuesta del Sistema
1. El cliente del sistema solicita una opción deseada.	1.1. El sistema a partir de la opción seleccionada se encarga de obtener la información. 1.2. El sistema muestra la pantalla con la información correspondiente.
Curso alternativo	

Tabla 15. Descripción del caso de uso del sistema <*Controlar navegación del sistema*>

CUS_4	Controlar navegación del sistema.
Actores	Cliente
Propósito	Permitir la navegación entre las pantallas.
Resumen	El caso de uso se inicia cuando el cliente pasa de una opción a otra para solicitar información.
Precondiciones	
Poscondiciones	El cliente solo podrá interactuar con una pantalla del tópico, la que corresponda a la opción seleccionada.
Curso Normal de los Eventos	
Acciones del Actor	Respuesta del Sistema
1. El cliente estando en una pantalla, solicita información que se encuentra en otra pantalla. 2. El cliente solicita información sobre un tópico seleccionado.	1.1. El sistema a partir de la selección realizada muestra la pantalla correspondiente. 2.1. El sistema muestra la pantalla con la información solicitada.
Curso alternativo	

Tabla 16. Descripción del caso de uso del sistema <*Presentar ayuda del sistema*>

CUS_5	Presentar ayuda del sistema.
Actores	Cliente
Propósito	Mostrar el contenido referido en esta opción.

Resumen	El caso de uso se inicia cuando el cliente solicita la opción de ayuda del sistema.	
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente estando en cualquier pantalla, solicita la opción de ayuda del sistema.		1.1. El sistema a partir de la solicitud realizada se encarga de obtener la información. 1.2. El sistema muestra la pantalla con la información correspondiente.
Curso alternativo		

Tabla 17. Descripción del caso de uso del sistema <*Admitir salida del cliente del sistema*>

CUS_6	Admitir salida del cliente del sistema.	
Actores	Cliente	
Propósito	Admitir la salida del sistema.	
Resumen	El caso de uso se inicia cuando el cliente solicita la salida del sistema.	
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente solicita la salida del sistema.		1.1. El sistema se encarga de finalizar la aplicación. 1.2. El sistema verifica si el cliente desea finalizar la salida.
Curso alternativo	1.2. a-) Si acepta, el sistema finaliza. 1.2. b-) Si no acepta el sistema sigue prestando funcionalidades.	

Tabla 18. Descripción del caso de uso del sistema <*Interactuar con medias del sistema*>

CUS_7	Interactuar con medias del sistema.	
Actores	Cliente	
Propósito	Permitir la realización de las opciones de control que brinda el sistema.	
Resumen	El caso de uso se inicia cuando el cliente solicita controlar las medias: ejecutar, pausar, detener y controlar curso de video o audio.	
CU Asociados	Controlar operaciones con video o audio. <<extend>>	
Precondiciones		
Poscondiciones	El cliente solo podrá ejecutar una de las opciones que brinda el sistema para la interacción con las medias.	
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente solicita la opción de ejecutar la media seleccionada.		1.1. El sistema se encarga de reconocer la media seleccionada y mostrarla en pantalla al cliente.
Curso alternativo		

Tabla 19. Descripción del caso de uso del sistema <*Controlar curso de video o audio*>

CUS_8	Controlar curso de video o audio. <<extend>>	
Actores	Cliente	
Propósito	Permitir la realización de la opción de control que brinda el sistema.	
Resumen	El caso de uso se inicia cuando el cliente solicita controlar el curso del video o audio.	
CU Asociados		
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente solicita la opción de controlar el		1.1. El sistema se encarga de realizar la

curso de la media seleccionada.	operación correspondiente a la media seleccionada.
Curso alternativo	

Tabla 20. Descripción del caso de uso del sistema <*Controlar las operaciones con video o audio*>

CUS_9	Controlar operaciones con video o audio. <<extend>>	
Actores	Cliente	
Propósito	Controlar la realización de las operaciones sobre las medias de video o audio, como son: ejecutar, pausar y detener.	
Resumen	El caso de uso se inicia cuando el cliente solicita realizar una operación de control sobre una media de tipo video o audio, ya sea detener, pausar o ejecutar.	
CU Asociados		
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente solicita la controlar o manipular la media seleccionada.		1.1. El sistema se encarga de realizar la operación correspondiente a la media seleccionada.
Curso alternativo	1.2. Si el cliente solicita ejecutar la media seleccionada, el sistema se encarga de la reproducción de la misma. 1.3. Si el cliente solicita pausar la media seleccionada, el sistema se encarga de pausarla para su posterior reproducción, tomando como punto inicial donde se detuvo. 1.4. Si el cliente solicita detener la media seleccionada, el sistema se encarga de detener la misma.	

Tabla 21. Descripción del caso de uso del sistema <*Obtener información de la BD*>

CUS_10	Obtener información de la BD. <<include>>	
Actores	Cliente	
Propósito	Permitir la realización de la obtención de información que brinda el sistema.	
Resumen	El caso de uso se inicia cuando el cliente solicita información sobre un tópico seleccionado o la ayuda.	
CU Asociados		
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente solicita información sobre un tópico seleccionado o la ayuda.		1.1. El sistema se encarga de obtener la información referida a lo que se ha seleccionado.
Curso alternativo		

Examen

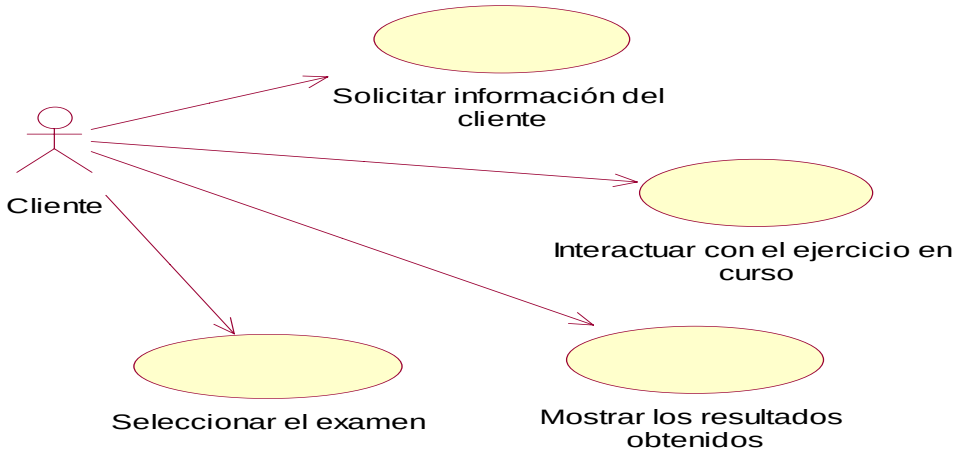


Figura 33. Diagrama de Caso de Uso de Examen

Referencia	Caso de Uso	Prioridad
CUS_ 11	Solicitar información del cliente.	Crítico
CUS_ 12	Seleccionar el examen.	Crítico

CUS_ 13	Interactuar con ejercicio en curso.	Crítico
CUS_ 14	Mostrar resultados obtenidos.	Crítico

Tabla 22. Descripción del caso de uso del sistema <*Solicitar información del cliente*>

CUS_11	Solicitar información del cliente.	
Actores	Cliente	
Propósito	Recopilar los datos del cliente.	
Resumen	El cliente inicia el caso de uso cuando solicita realizar un test, aquí el sistema debe recoger los datos necesarios del cliente.	
CU Asociados		
Precondiciones	El cliente debe registrar los datos que se le piden, de lo contrario no podrá realizar el test.	
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente realiza la entrada de datos correspondientes, previos a seleccionar un test.		1.1. El sistema se encarga de recopilar los datos del cliente para brindar una información personalizada.
Curso alternativo		

Tabla 23. Descripción del caso de uso del sistema <*Seleccionar el examen*>

CUS_12	Seleccionar el examen.	
Actores	Cliente	
Propósito	Dar la posibilidad de seleccionar el test a realizar.	
Resumen	El cliente inicia el caso de uso cuando realiza la selección del test que va a realizar.	
CU Asociados		
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema

1. El cliente solicita realizar un test. 2. El cliente selecciona el test a realizar.	1.1. El sistema se encarga de posibilitar una selección entre los diversos test que existen. 2.1. El sistema se encarga de mostrar la pantalla correspondiente con el test seleccionado.
Curso alternativo	

Tabla 24. Descripción del caso de uso del sistema <*Interactuar con ejercicio en curso*>

CUS_13	Interactuar con ejercicio en curso.
Actores	Cliente
Propósito	Posibilitar que el cliente manipule el ejercicio.
Resumen	El cliente inicia el caso de uso cuando comienza a interactuar con el ejercicio seleccionado.
CU Asociados	
Precondiciones	
Poscondiciones	
Curso Normal de los Eventos	
Acciones del Actor	Respuesta del Sistema
1. El cliente interactúa con el ejercicio para la realización del mismo.	1.1. El sistema se encarga de que las opciones brindadas para la solución del ejercicio, sean manipulables.
Curso alternativo	

Tabla 25. Descripción del caso de uso del sistema <*Interactuar con ejercicio en curso*>

CUS_14	Mostrar resultados obtenidos.
Actores	Cliente
Propósito	Dar la posibilidad de mostrar de una manera personalizada los resultados correspondientes al test realizado.
Resumen	El cliente inicia el caso de uso cuando comienza a interactuar con el ejercicio seleccionado.

CU Asociados		
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente finaliza el test seleccionado.		1.1. El sistema se encarga de finalizar el test realizado. 2.1. El sistema se encarga de mostrar los resultados correspondientes a la realización del test seleccionado, con una información personalizada del cliente.
Curso alternativo		

Galería

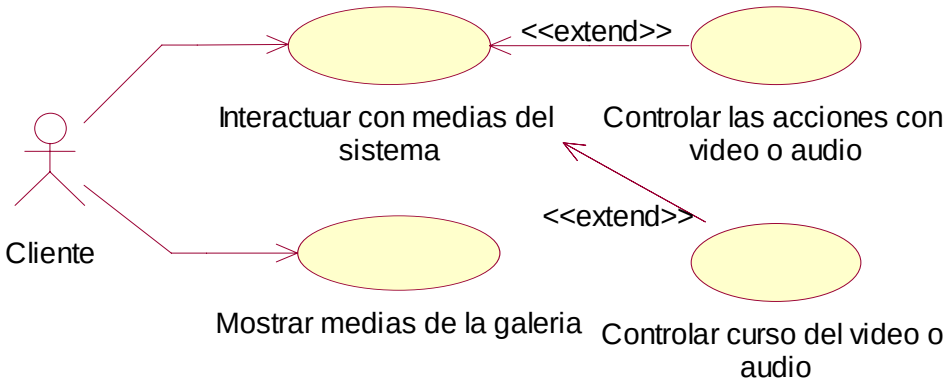


Figura 34. Diagrama de Caso de Uso de Galería

Referencia	Caso de Uso	Prioridad
CUS_ 15	Mostrar medias de la galería.	Crítico

Tabla 26. Descripción del caso de uso del sistema <*Mostrar medias de la galería*>

CUS_15	Mostrar medias de la galería.
Actores	Cliente

Propósito	Dar la posibilidad de mostrar las medias de la galería.
Resumen	El cliente inicia el caso de uso cuando solicita una determinada media de la galería.
CU Asociados	Interactuar con medias del sistema.
Precondiciones	
Poscondiciones	

Curso Normal de los Eventos	
Acciones del Actor	Respuesta del Sistema
1. El cliente solicita media de la galería. 2. El cliente selecciona media de la galería. 3. El cliente interactúa con la media seleccionada.	1.1. El sistema se encarga de mostrar las medias de la galería. 2.1. El sistema se encarga de visualizar la media seleccionada de la galería en la pantalla correspondiente. 3.1. El sistema se encarga de realizar los controles necesarios a la media seleccionada.

Curso alternativo	

Glosario

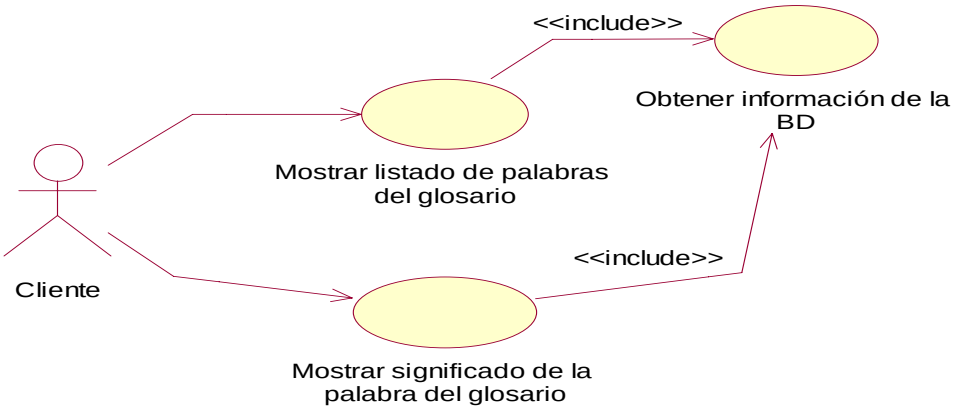


Figura 35. Diagrama de Caso de Uso de Glosario

Referencia	Caso de Uso	Prioridad
CUS_ 16	Mostrar listado de palabras del glosario.	Crítico
CUS_ 17	Mostrar significado de la palabra del glosario.	Crítico

Tabla 27. Descripción del caso de uso del sistema <*Mostrar listado de palabras del glosario*>

CUS_16	Mostrar listado de palabras del glosario.	
Actores	Cliente	
Propósito	Mostrar el listado de palabras que componen el glosario.	
Resumen	El cliente inicia el caso de uso cuando solicita las palabras del glosario.	
CU Asociados	Obtener información de la base de datos.	
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente solicita listado de palabras que componen el glosario.		1.1. El sistema se encarga de obtener la información correspondiente. 1.2. El sistema se encarga de mostrar la pantalla correspondiente con el listado de palabras.
Curso alternativo		

Tabla 28. Descripción del caso de uso del sistema <*Mostrar significado de la palabra del glosario*>

CUS_17	Mostrar significado de la palabra del glosario.
Actores	Cliente
Propósito	Mostrar el significado correspondiente a la palabra seleccionada.
Resumen	El cliente inicia el caso de uso cuando solicita el

	significado de la palabra seleccionada del glosario.	
CU Asociados	Obtener información de la base de datos.	
Precondiciones		
Poscondiciones		
Curso Normal de los Eventos		
Acciones del Actor		Respuesta del Sistema
1. El cliente solicita el significado de la palabra seleccionada.		1.1. El sistema se encarga de obtener la información correspondiente. 1.2. El sistema se encarga de mostrar la pantalla con el significado de la palabra seleccionada.
Curso alternativo		

Diagramas de Presentación del Modelo del Diseño

Este es un artefacto nuevo dentro del lenguaje UML, incorporado a este a partir de la extensión del mismo planteada por OMMMA-L, el cual sirve para describir la parte estática del modelo a través de una descripción intuitiva de la distribución espacial de objetos visuales de la interfaz de usuario. OMMMA-L utiliza los diagramas de presentación y modifica los diagramas de clases, este último se divide en dos áreas: una para la jerarquía de los tipos de media y otra para la modelación de la estructura lógica del dominio de la aplicación.

Diagrama de Presentación General

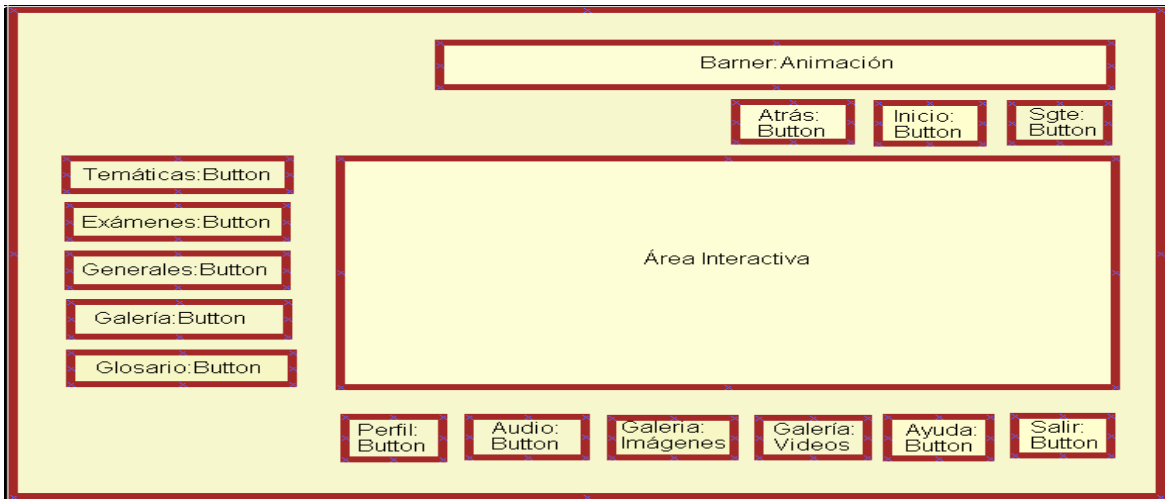


Figura 36. Diagrama de Presentación General

Diagrama de Presentación Galería de Imágenes

Figura 37. Diagrama de Presentación Galería de Imágenes

Diagrama de Presentación Galería de Videos

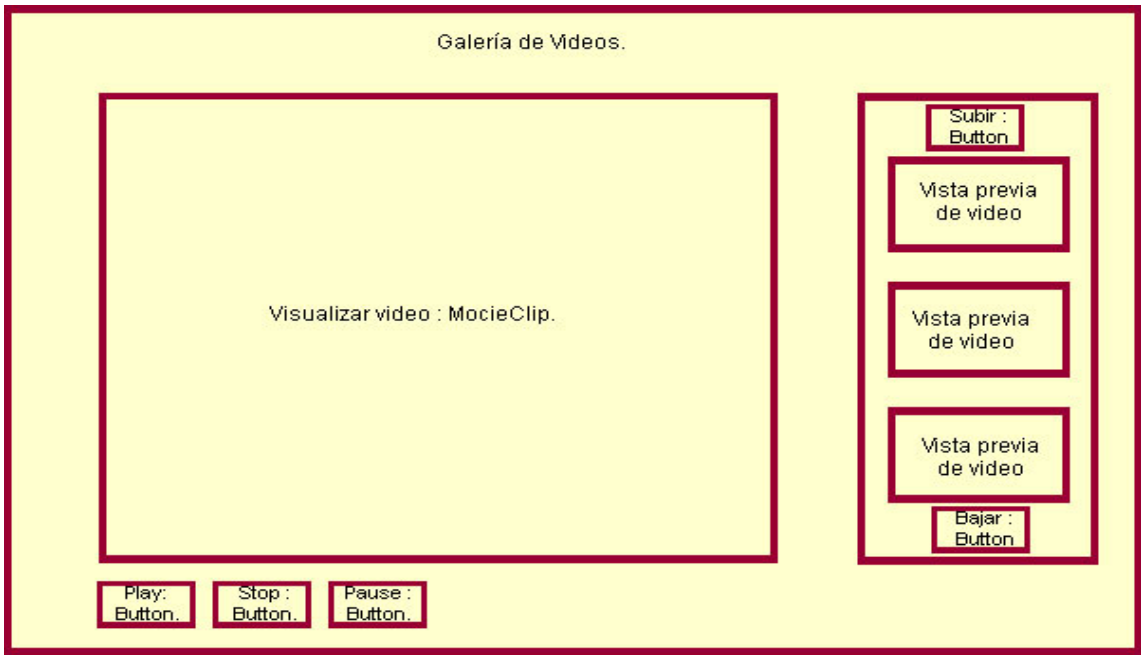


Figura 38. Diagrama de Presentación Galería de Videos.

Diagrama de Presentación Glosario

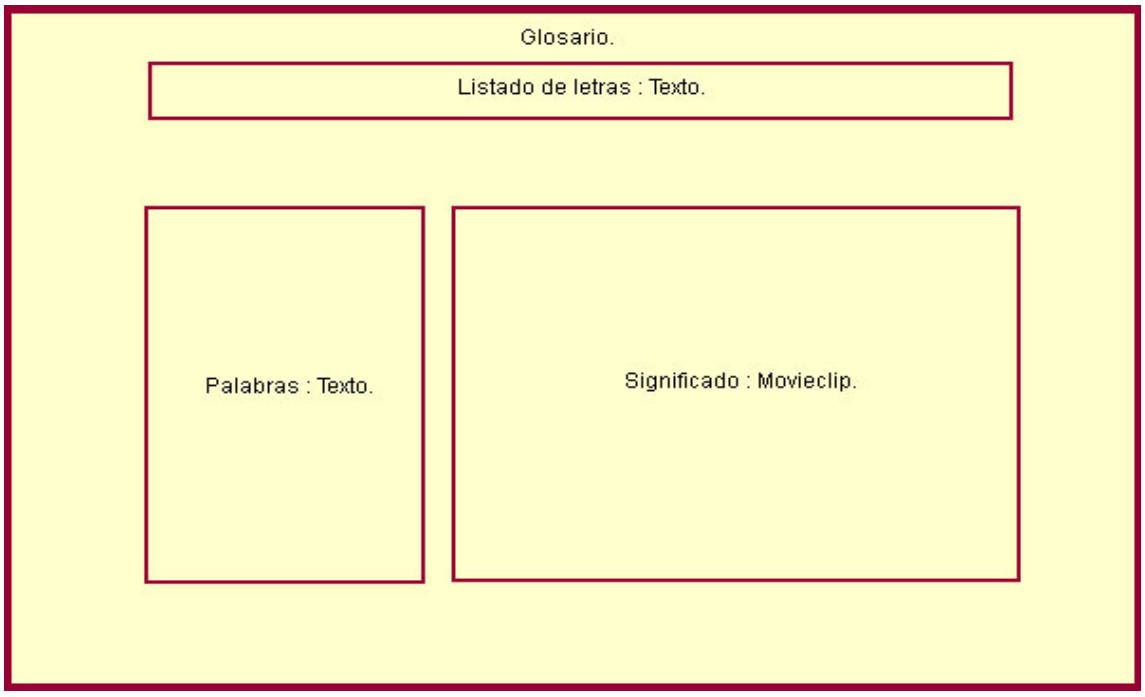


Figura 39. Diagrama de Presentación Glosario.

Diagrama de Presentación Examen

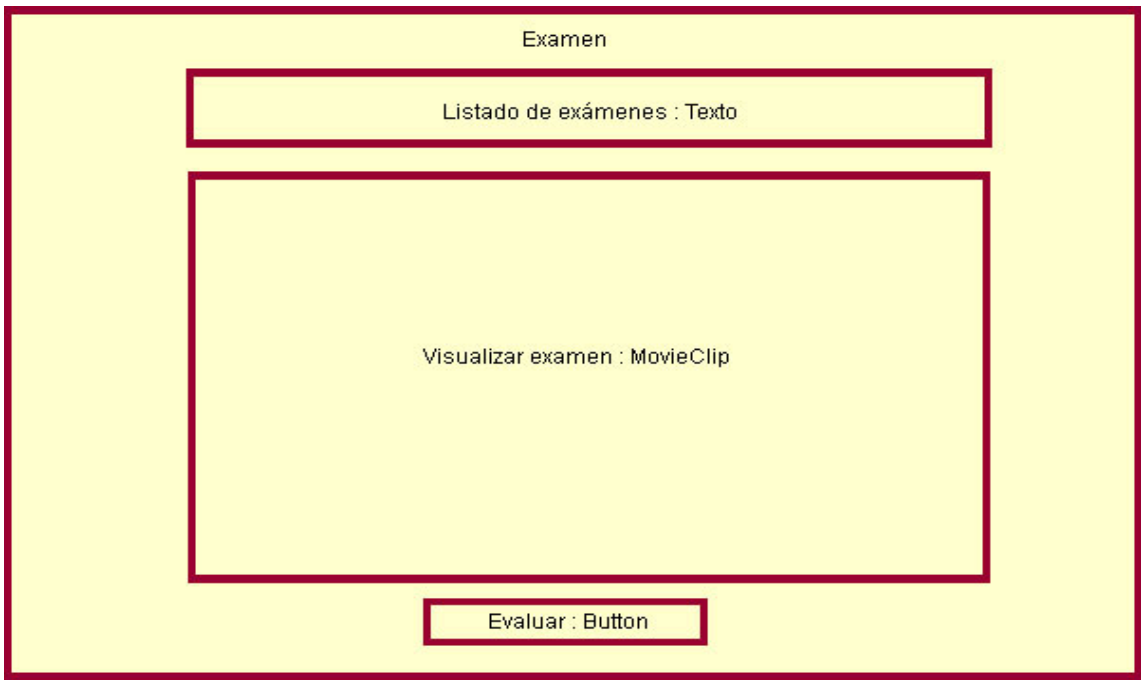


Figura 40. Diagrama de Presentación Examen.

Diagrama de Jerarquía de Clases

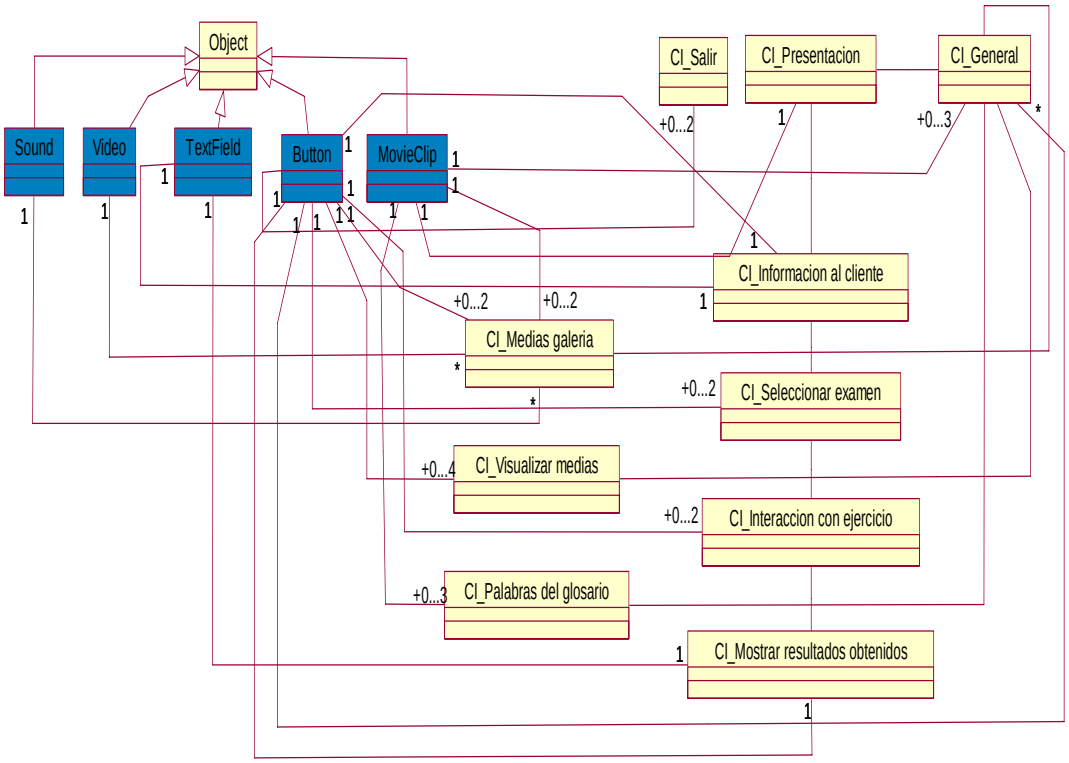


Figura 41. Diagrama Jerárquico de las Clases.

Modelo del Diseño

Diagrama de Clases del Paquete Presentación

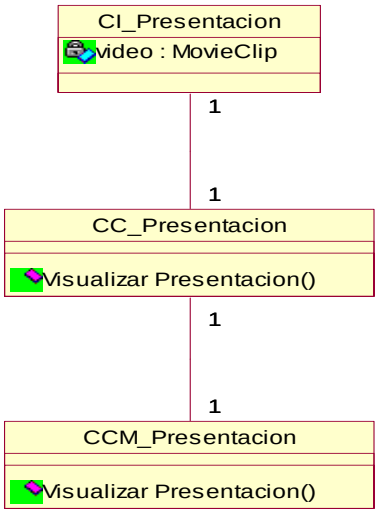


Figura 42. Diagrama de Clases del Paquete Presentación.

Diagrama de secuencia del casos de uso relacionado con Presentación <*Cargar presentación del sistema*>

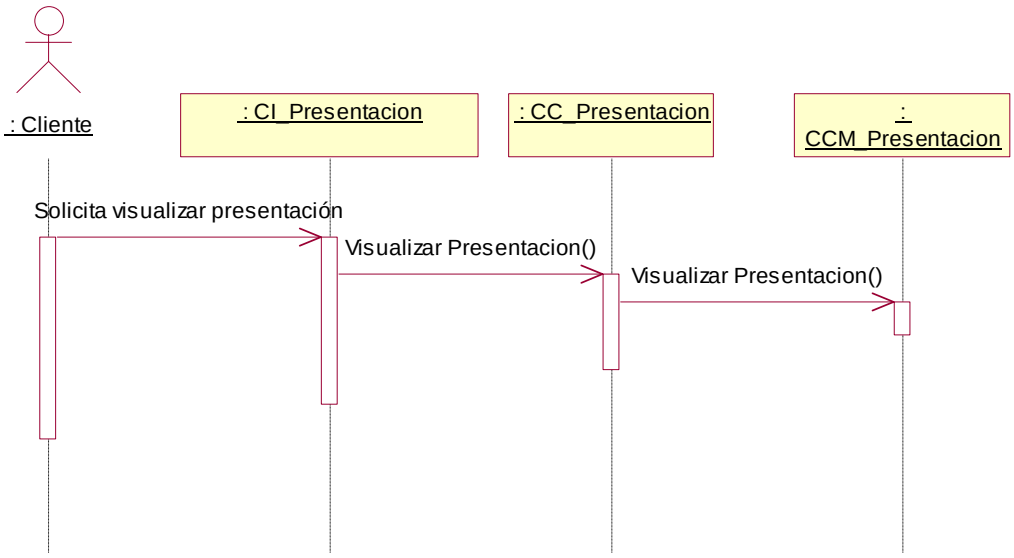


Diagrama de Clases del Paquete Galería

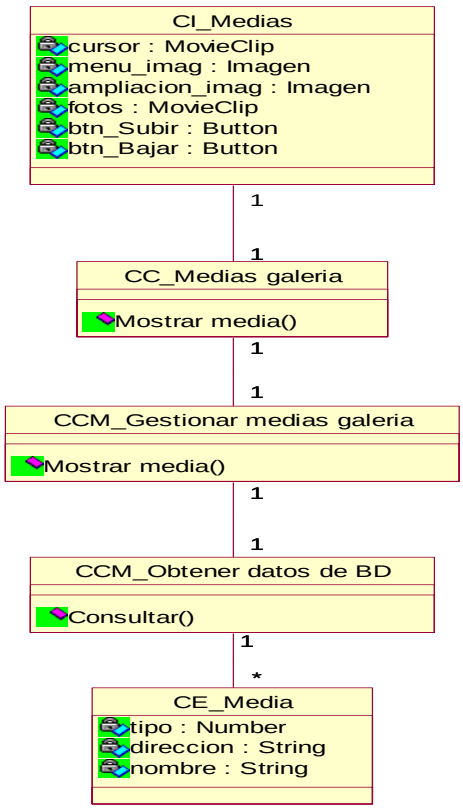


Figura 43. Diagrama de Clases del Paquete Galería.

Diagrama de secuencia del casos de uso relacionado con Galería <*Mostrar medias de la galería*>

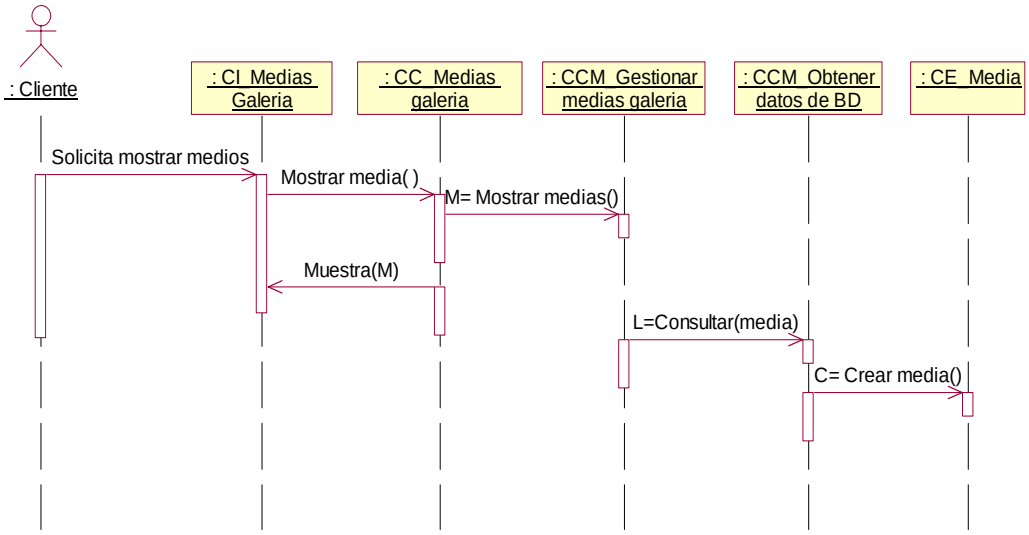
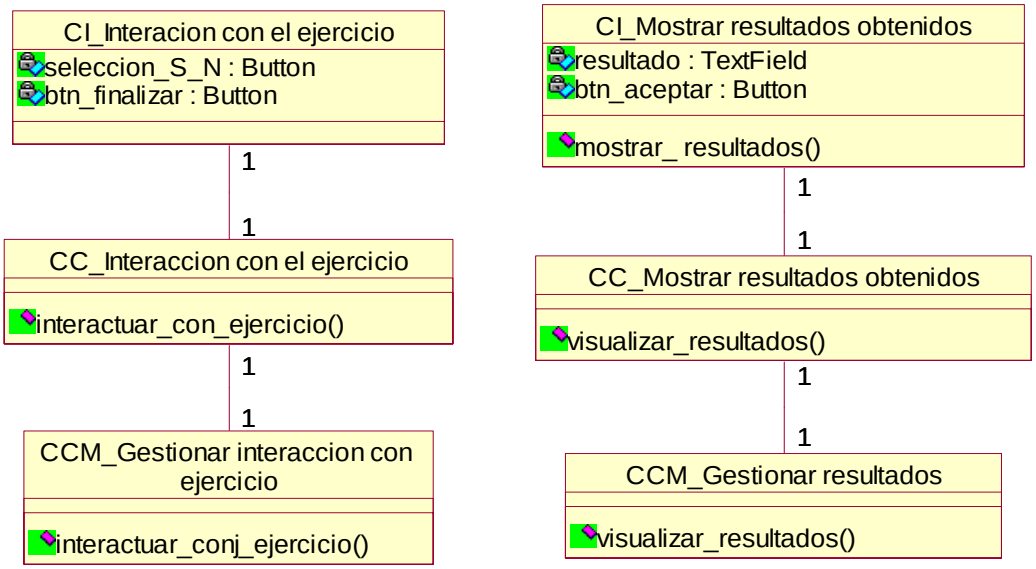


Diagrama de Clases del Paquete Examen



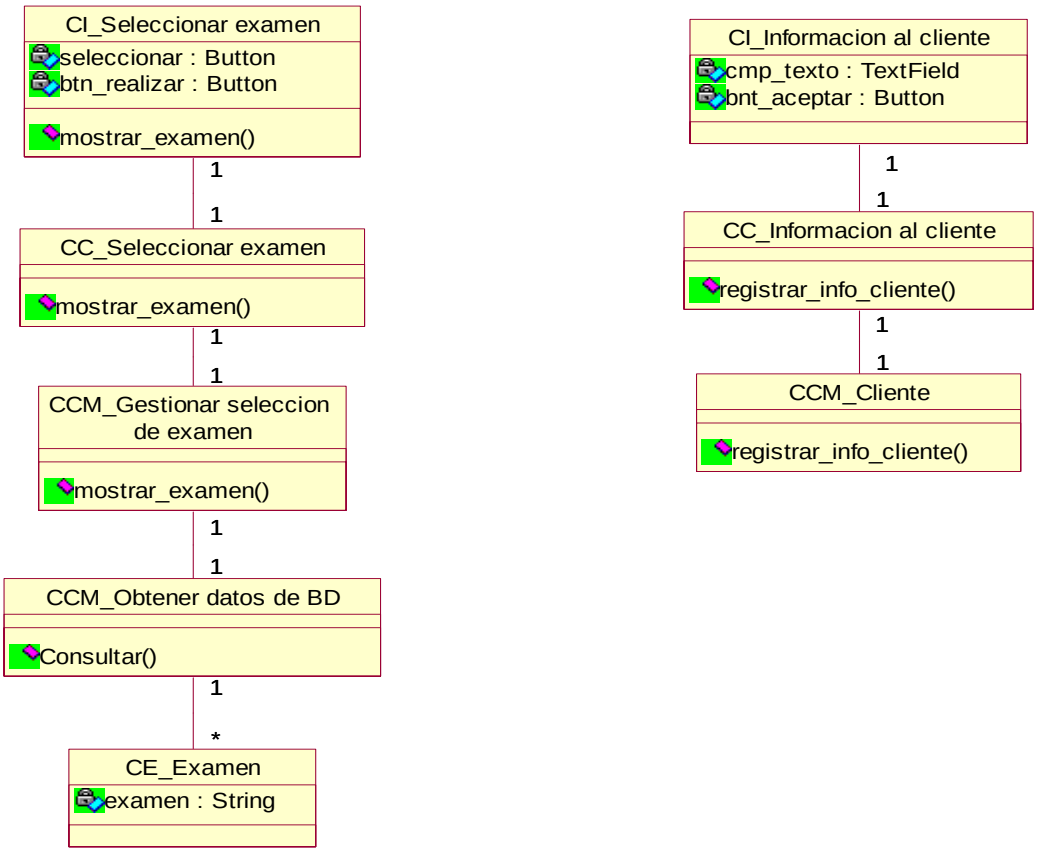


Figura 44. Diagrama de Clases del Paquete Examen.

Diagramas de Secuencia de los Casos de Uso relacionados con Examen

- Solicitar información del cliente
- Seleccionar examen
- Interactuar con ejercicio en curso
- Mostrar resultados obtenidos en el examen

Diagrama de Secuencia <*Solicitar información del cliente*>

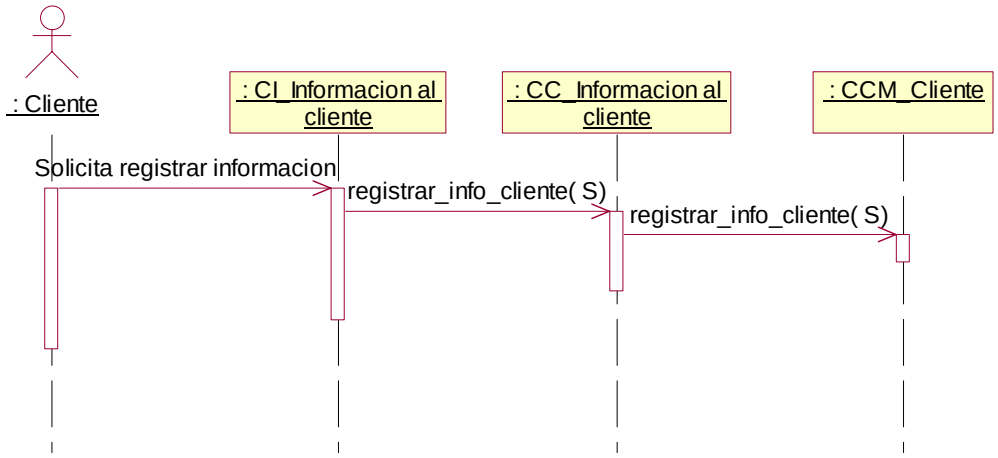


Diagrama de Secuencia <*Seleccionar examen*>

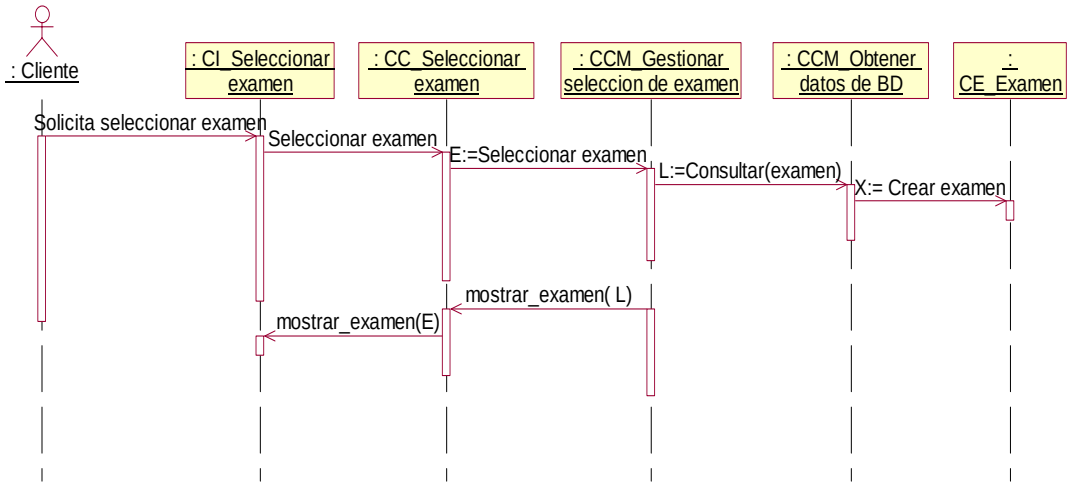


Diagrama de Secuencia <*Interactuar con ejercicio en curso*>

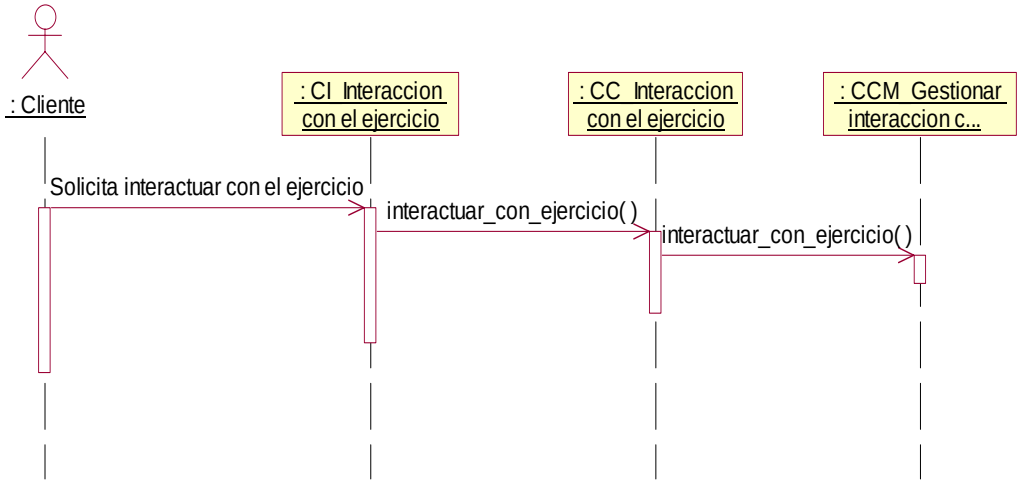


Diagrama de Secuencia <*Mostrar resultados obtenidos*>

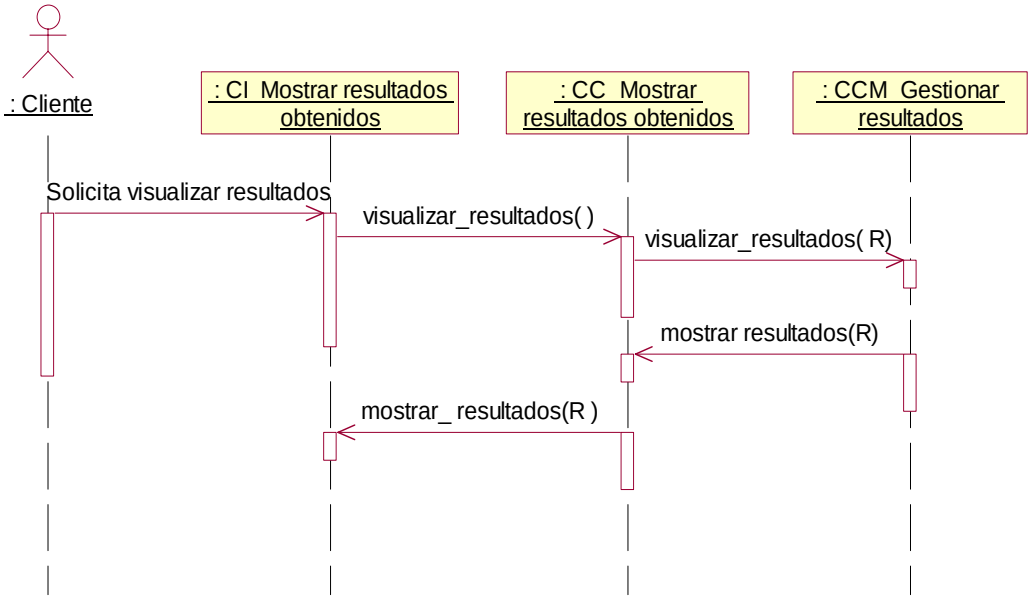


Diagrama de Clases del Paquete Generales

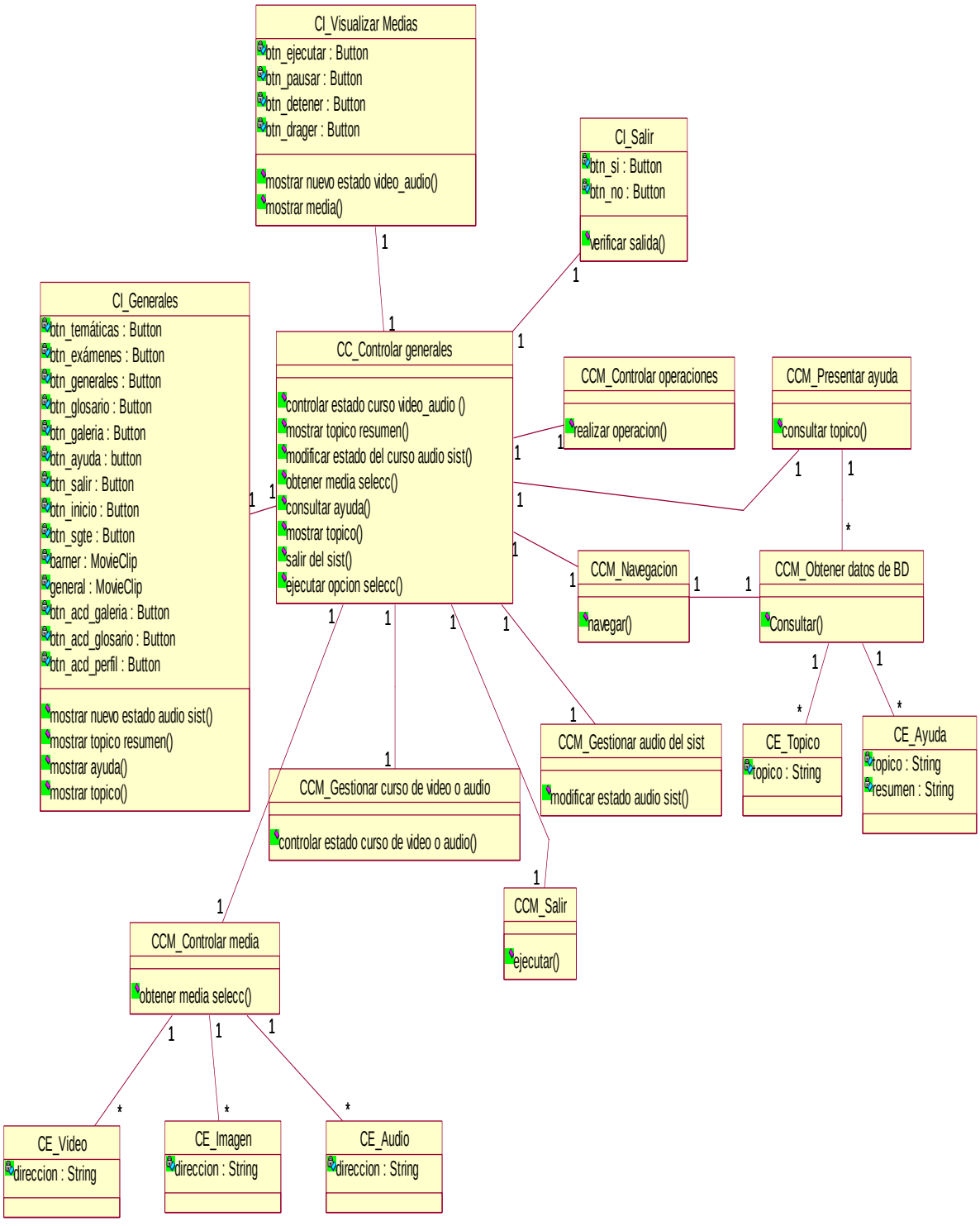


Figura 45. Diagrama de Clases del Paquete Generales.

Diagramas de Secuencia de los Casos de Uso relacionados con Generales

• Controlar audio del sistema.

• Controlar curso de video o audio.

• Controlar navegación del sistema.

• Controlar operaciones con video o audio.

• Interactuar con medias del sistema.

• Presentar ayuda del sistema.

• Mostrar contenido del tema seleccionado.

• Admitir salida del cliente del sistema.

Diagrama de Secuencia <*Controlar audio del sistema*>

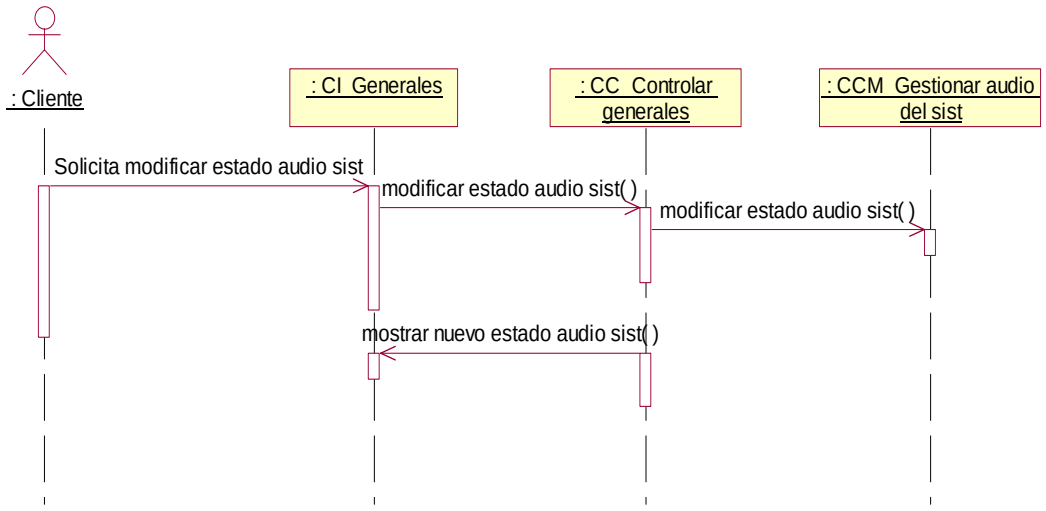


Diagrama de Secuencia <*Controlar curso de video o audio*>

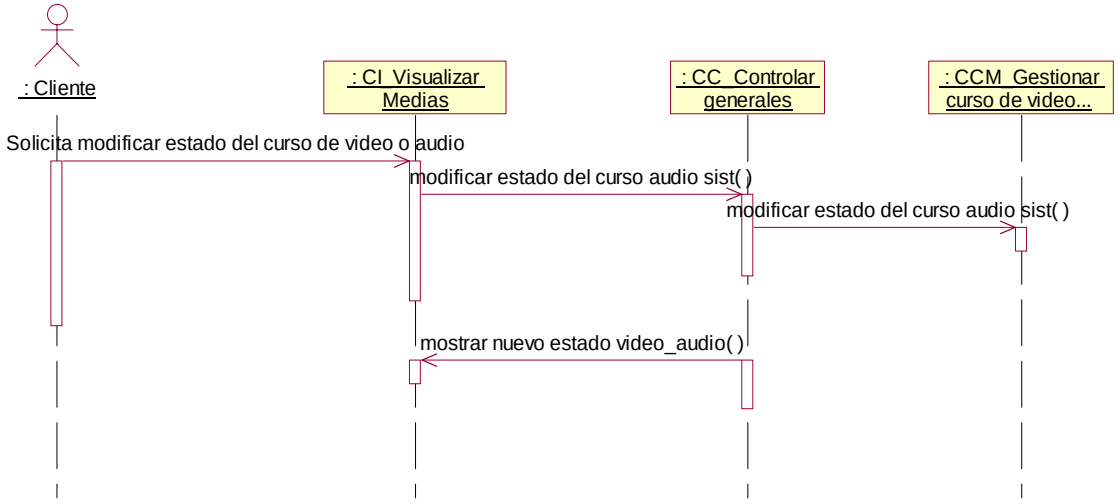


Diagrama de Secuencia <*Controlar navegación del sistema*>

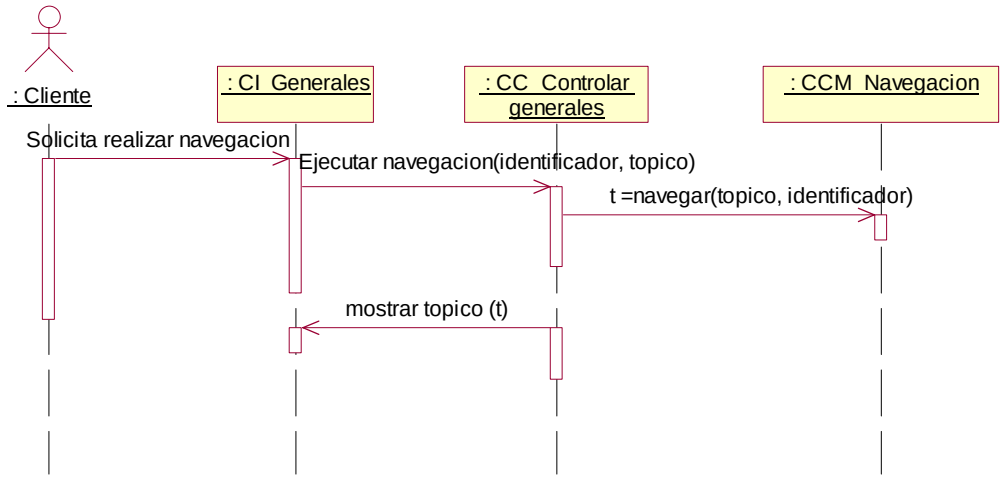


Diagrama de Secuencia <*Controlar operaciones con video o audio*>

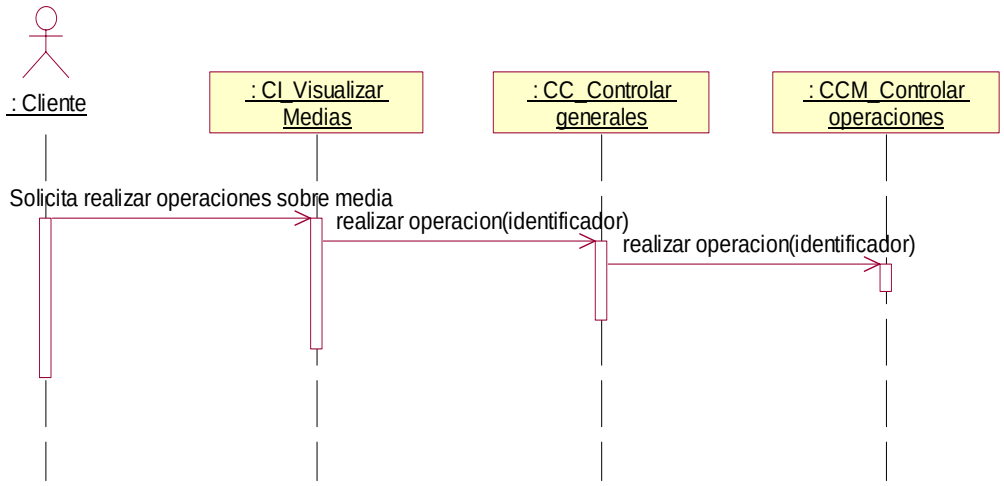


Diagrama de Secuencia <*Interactuar con medias del sistema*>

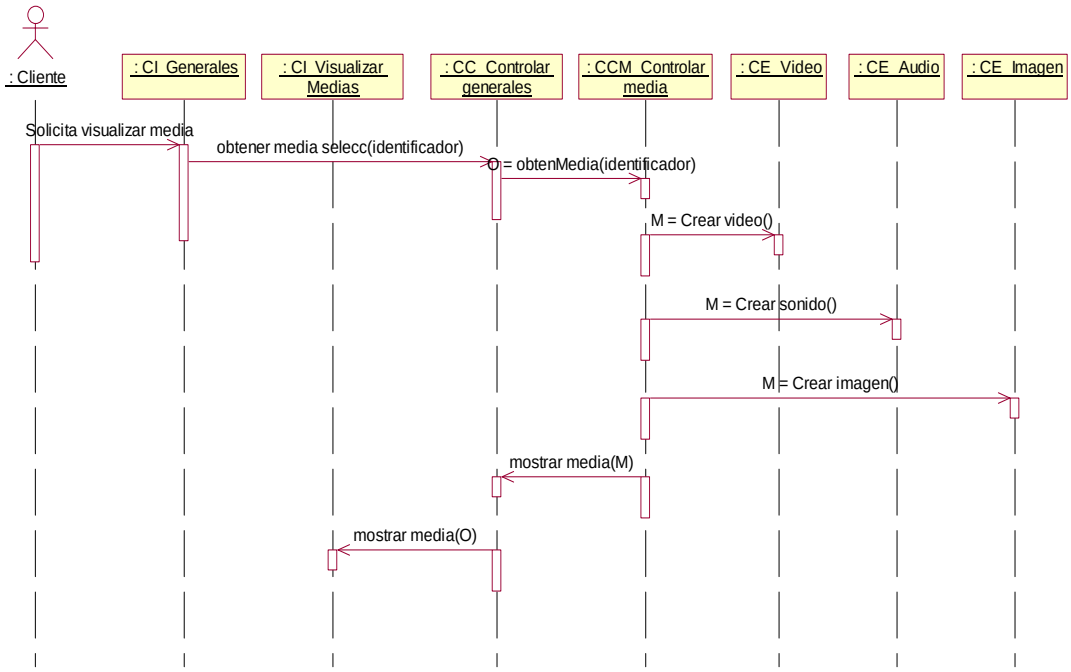


Diagrama de Secuencia <*Presentar ayuda del sistema*>

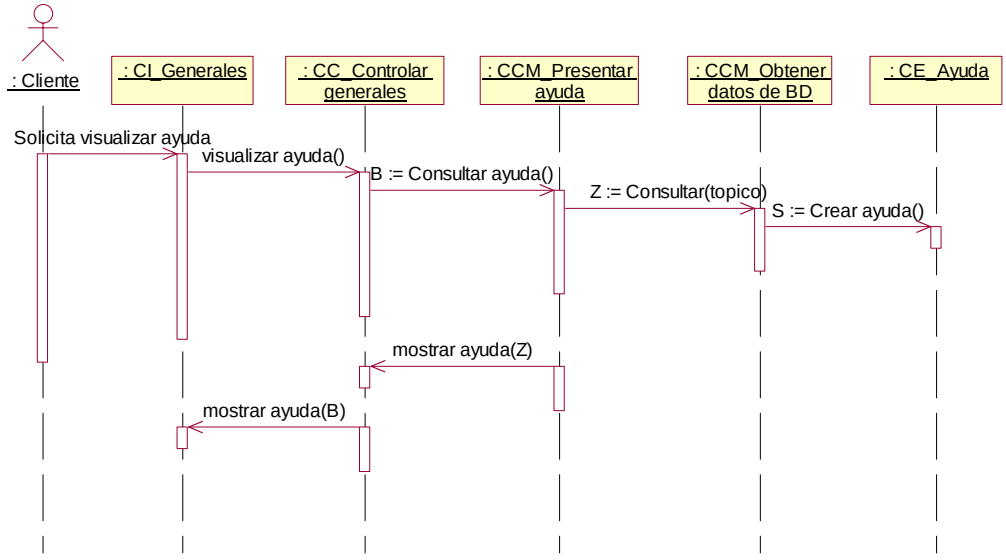


Diagrama de Secuencia <*Mostrar contenido del tópico seleccionado*>

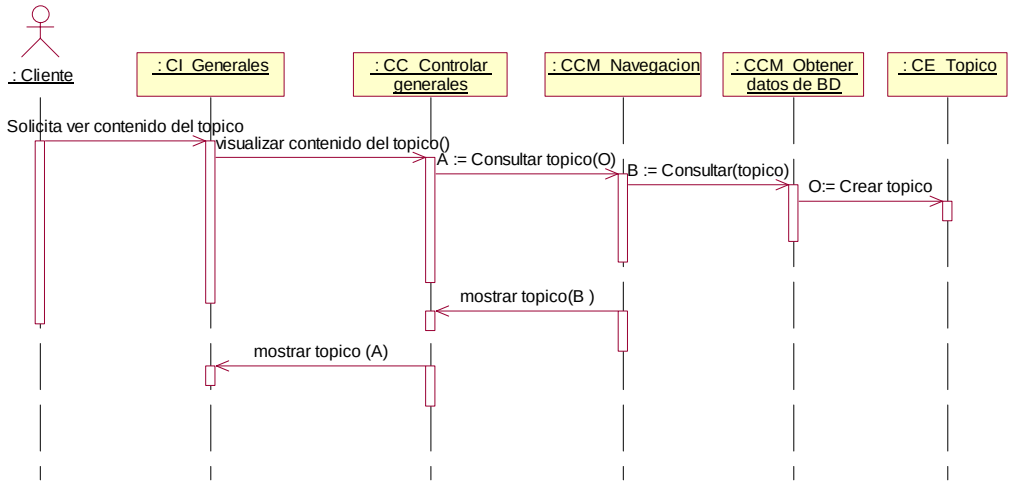


Diagrama de Secuencia <*Admitir salida del cliente del sistema*>

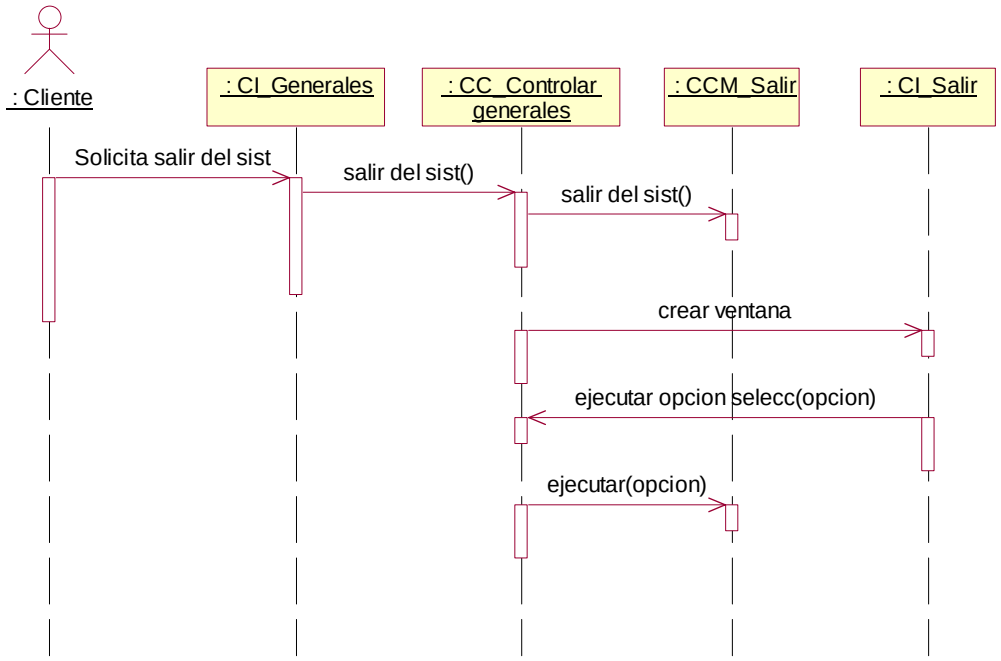


Diagrama de Clases del Paquete Glosario

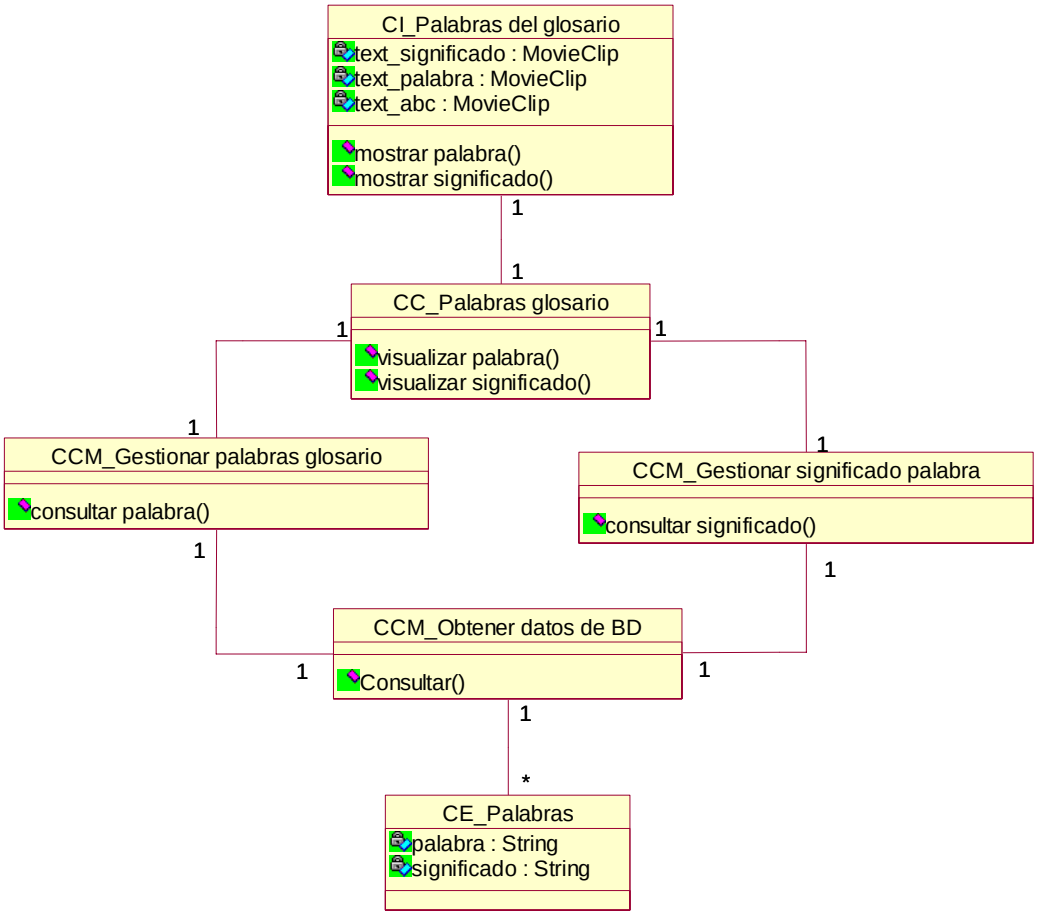


Figura 46. Diagrama de Clases del Paquete Glosario.

Diagramas de Secuencia de los Casos de Uso relacionados con Glosario

- Mostrar listado de palabras del glosario
- Mostrar significado de palabras del glosario

Diagrama de Secuencia <*Mostrar listado de palabras del glosario*>

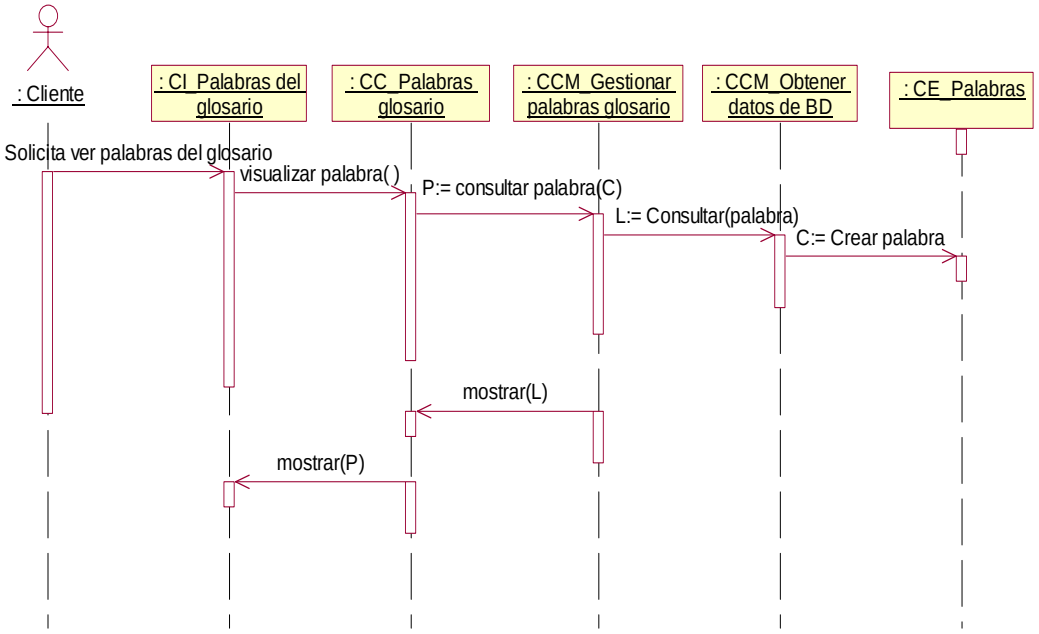
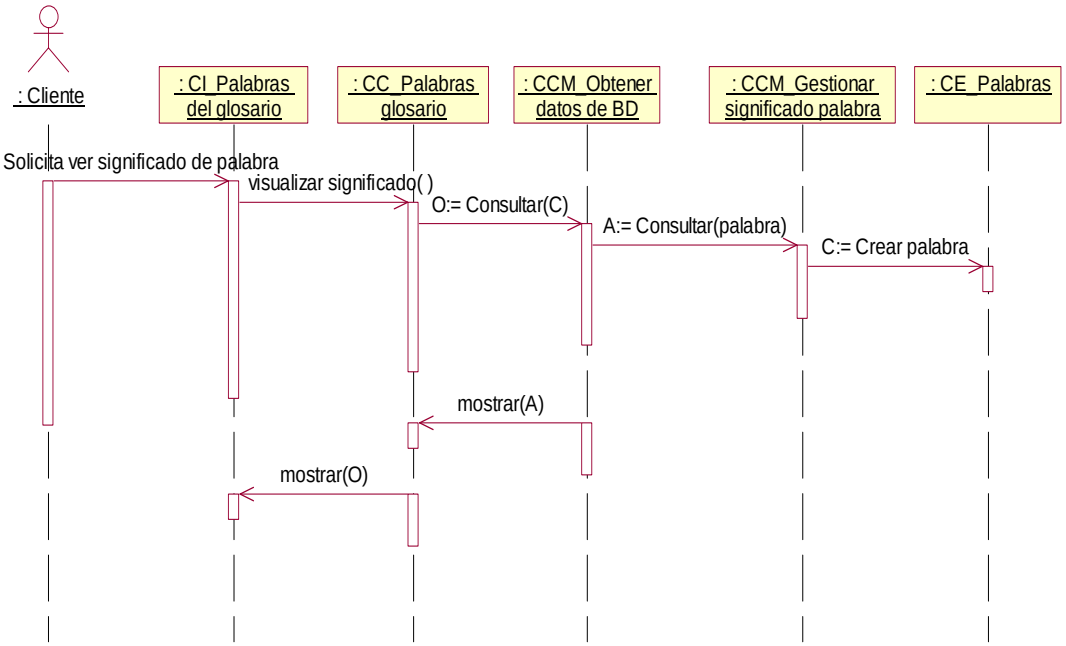


Diagrama de Secuencia <*Mostrar significado de palabras del glosario*>



Diseño de la Base de Datos

Para realizar el diseño de la base de datos del sistema, se utilizó el diagrama de clases persistentes, algunas de las clases representan los datos que se obtienen y almacenan durante los procesos de la aplicación, lo que permitirá ver la relación entre los datos.

Diagrama de Clases Persistentes

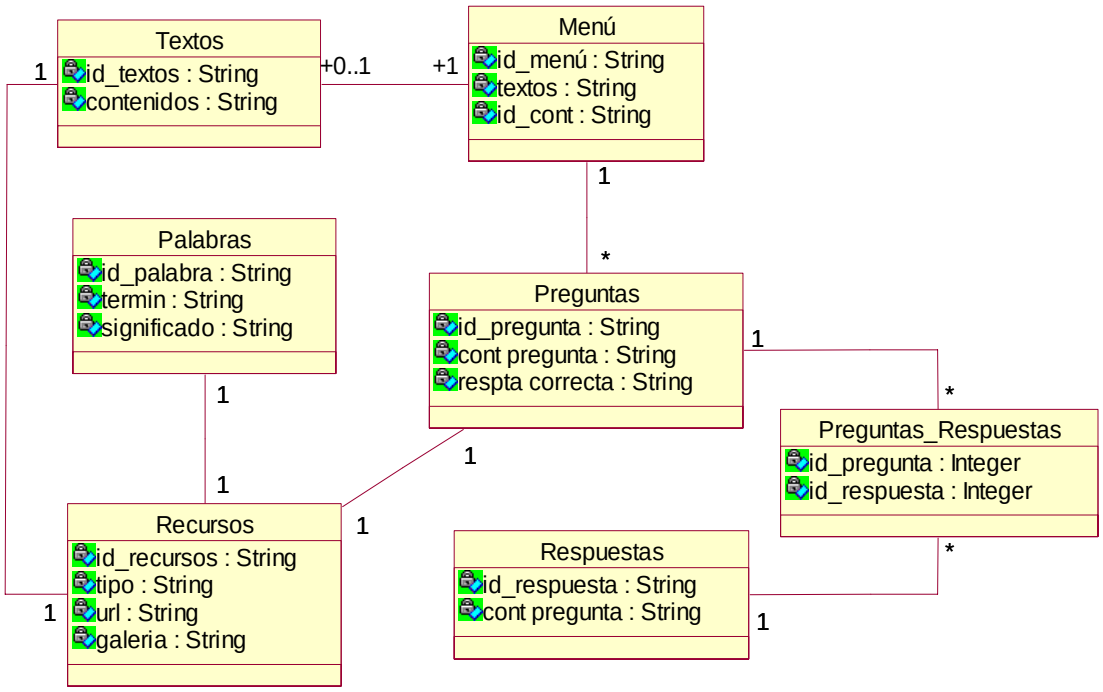


Figura 47. Diagrama de Clases Persistentes.

Modelo de Datos

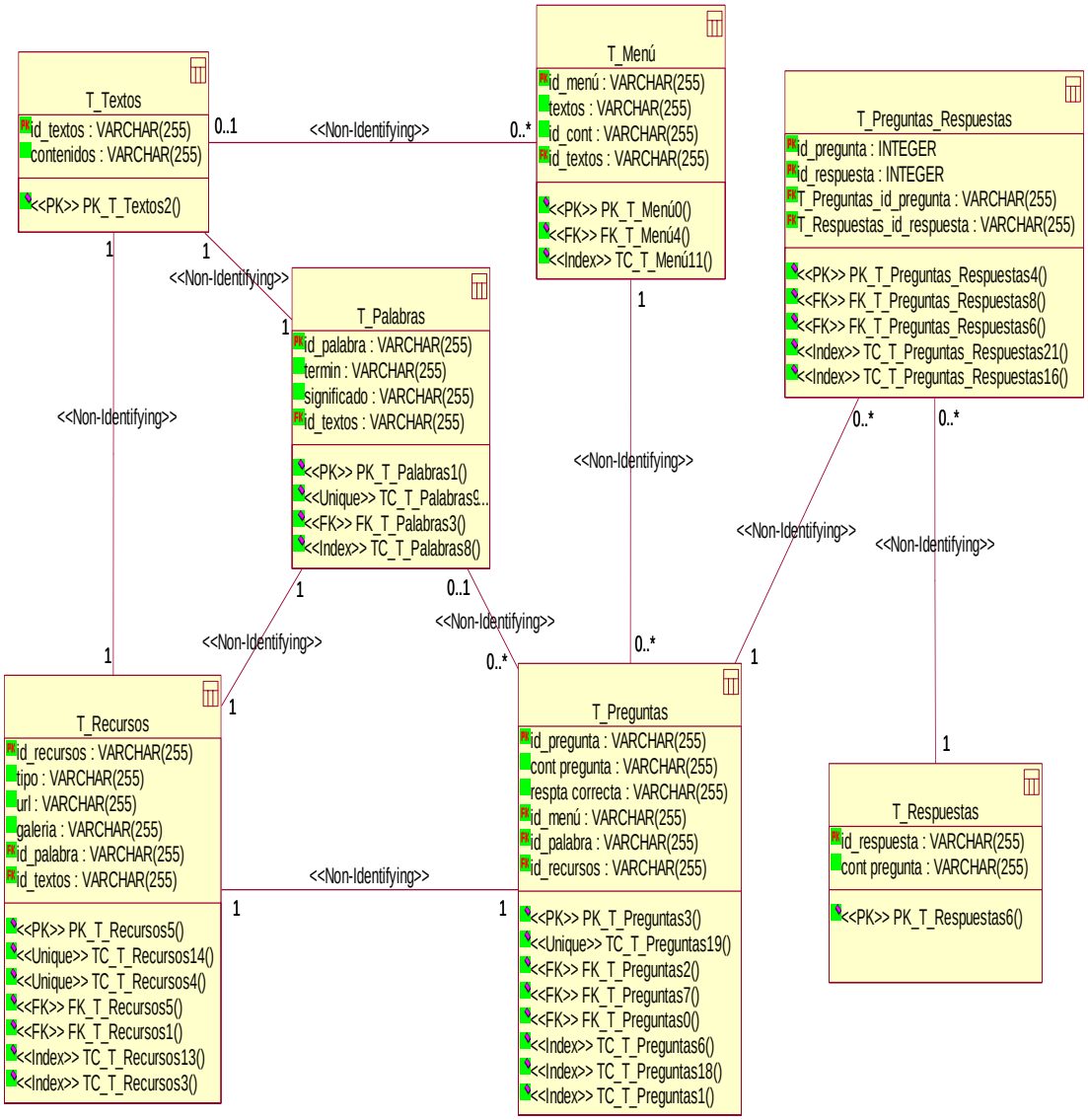


Figura 48. Modelo Lógico de los Datos.

Descripción de las tablas de la Base de Datos

Tabla 29. Descripción de la tabla Glosario

Nombre	Glosario	
Descripción	En esta tabla se almacenan todas las palabras y sus significados, que puedan resultar de interés para el cliente.	
Atributo	Tipo	Descripción

id_palabra	String	Identificador de la tabla.
termino	String	Palabra seleccionada.
significado	String	Significado de la palabra.

Tabla 30. Descripción de la tabla Preguntas

Nombre	Preguntas	
Descripción	En esta tabla se almacenan todas las preguntas que conforman los test.	
Atributo	Tipo	Descripción
Id_pregunta	String	Identificador de la tabla.
cont_pregta	String	Pregunta del ejercicio
resp_correcta	String	Respuesta correcta del ejercicio

Tabla 31. Descripción de la tabla Respuestas

Nombre	Respuestas	
Descripción	En esta tabla se almacenan todas las posibles respuestas de los test.	
Atributo	Tipo	Descripción
id_respuesta	String	Identificador de la tabla.
cont_respuesta	String	Pregunta del ejercicio

Modelo de Implementación

Diagrama de Paquetes

Figura 49. Diagrama de Paquetes.

Diagrama de Componentes

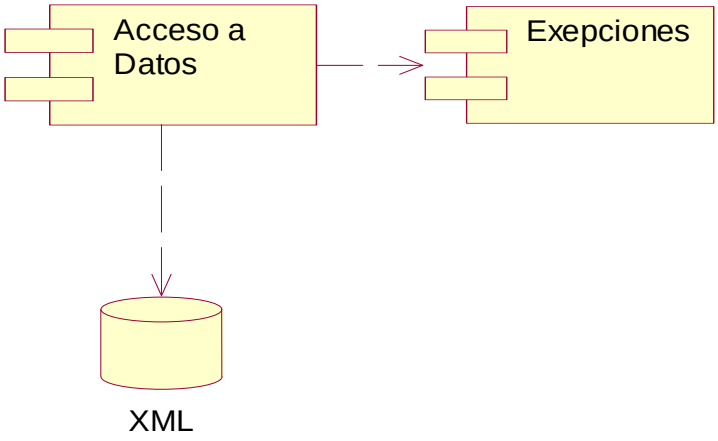


Figura 50. Diagrama de Componentes del Paquete BD_Inglés.

Diagrama de Componentes del paquete Modelo de Implementación

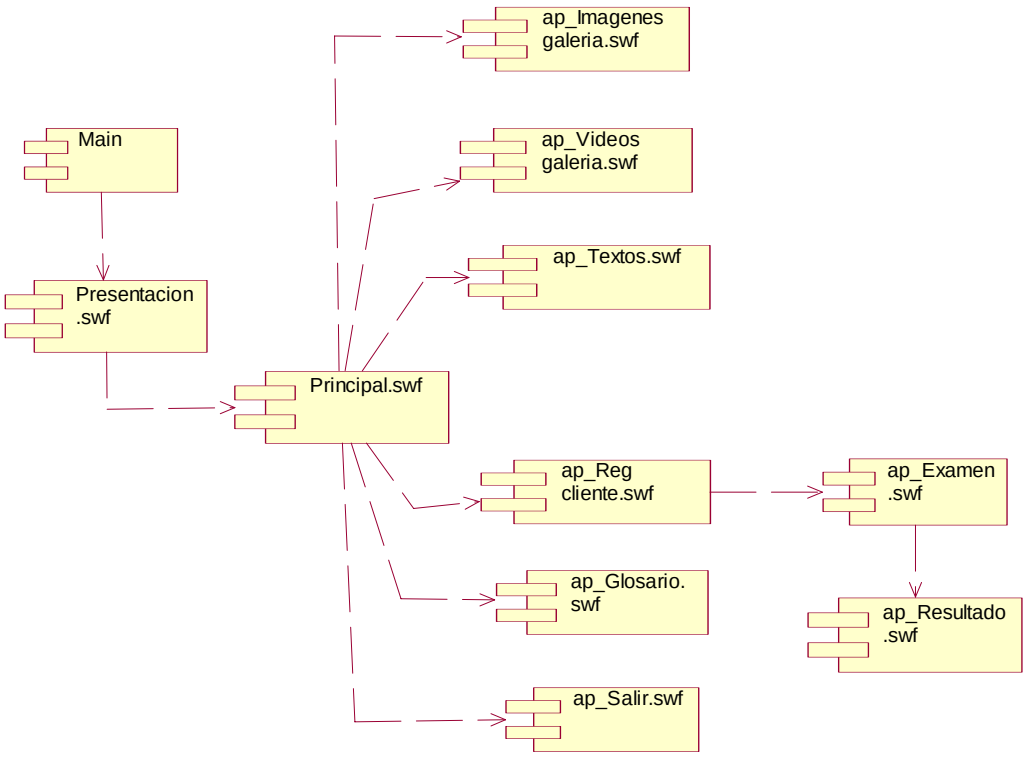


Figura 50. Diagrama de Componentes del Paquete Modelo de Implementación.

Conclusiones

Con el presente Trabajo de Diploma se provee al ISMM de una guía metodológica de RUP, la que brinda una importante documentación para los planes de estudio de la carrera de Informática, permitiendo una mejor asimilación de este contenido en la asignatura de Ingeniería de Software.

A partir del trabajo realizado se obtuvo la propuesta de adaptación de RUP para grupos de trabajo pequeños, la cual recoge los roles y las actividades que estos realizan, según los flujo de trabajo, así como los artefactos que se generan. Esta propuesta es adaptable según el tipo de aplicación (Escritorio, Web, Multimedia).

También se realizó un ejemplo de RUP adaptado para cada tipo de aplicación que ayuda en la utilización esta adaptación.

Recomendaciones

En esta investigación se logran los objetivos planteados, y se recomienda su utilización para lograr una mejor incursión en proyectos con grupos de trabajo de pocos integrantes.

Para ratificar los resultados de esta guía:

- Realizar modificaciones en los flujos de trabajo de apoyo de RUP.
- Se propone realizar talleres para desarrolladores de software donde se tome en cuenta la adaptación de RUP en proyectos de trabajo pequeño.
- Utilizar la guía como material de estudio en la asignatura de Ingeniería de Software.

Referencias Bibliográficas

[1] A. Báez; C. Castañeda; D. Castañeda. Relación con RUP. [En línea].En su: Metodología para el Diseño y Desarrollo de Interfaces de Usuario. [Consultado: 2007/12/03]. Disponible en: http://pegasus.javeriana.edu.co/~fwj2ee/descargas/metodologia(v0.1).pdf

[2] A. Cockburn, Crystal/Clear: A Human-Powered Methodology for Small Teams, to be published by Addison- Wesley, Reading, Mass., 2000; access our early version at members.aol.

[3] Dean Leffingwell; Don Widrig; Addison Wesley. Managing Software Requirements, First Edition, 1999.

[4] Foros del Web. Anomalías en los proyectos intensivos de software. [En línea]. [Consultado: 2008/01/16]. Disponible en: http://www.forosdelweb.com

[5] Groups. Tarjeta CRC. [En línea]. [Consultado: 2008/02/20]. Disponible en: http://www. groups.msn.com/UML/_messageboard.msnw

[6] Hipertexto. Mapas de Navegación. [En línea]. [Consultado: 2008/02/27]. Disponible en: http://www.hipertexto.info/documentos/maps_navegac.htm

[7] Jacobson Ivar, Booch Grady, Rumbaugh James. El Proceso Unificado de Desarrollo de Software.

[8] Kinetia. Proceso Unificado del Rational (RUP). [En línea]. [Consultado: 2007/11/05]. Disponible en: http://www.kynetia.es/calidad/rup-rational-unified-process.html

[9] Kweku Ewusi-Mensah. Software Development Failures: Anatomy of Abandoned Projects, 2003. [En línea] [Consultado: 2008/02/03]. Disponible en: http://books.google.com.bo/books?hl=es&id=cWde_yxJorEC&dq=Software+Development+Failures:+Anatomy+of+Abandoned+Projects+pdf&printsec=frontcover&source=web&ots=WaOrXolJF2&sig=p9i69ylKvWEGt3QcNhZ7926s5O0#PPR10,M1

[10] Larman C. *UML y Patrones.* Prentice Hall, México, 1999.

[11] Liz Augustine. Using the Rational Unified Process (RUP) Successfully for Small Development Projects. The Rational Edge. [En línea] [Consultado: 2007/11/07]. Disponible en: http://www.therationaledge.com/content/sep_01/m_RUPForSmall_la.html

[12] Microsoft. Los Ciclos y Etapas de RUP. [En línea]. [Consultado: 2007/11/07]. Disponible en: http://www.microsoft.com/spanish/msdn/arquitectura/roadmap_arq/heterodox.asp

[13] Philippe Kruchten, Rational Unified Process— An Introduction, Addison-Wesley, 1999.

[14] Portal. Desarrollo de la Navegación en entornos Web. [En línea]. [Consultado: 2008/02/27]. Disponible en: http://www.portal.huascaran.edu.pe/modulos/m_taller/mapa.htm

[15] Pressman, Ingeniería de Software. Un enfoque práctico. Quinta Edición. McGraw Hill Interamericana.

[16] Proyecto Web. Diseñando. Mapas de Navegación. [En línea]. [Consultado: 2008/02/27]. Disponible en: http://www.proyectoweb.org/boletin/cartografia-web-disenando-mapas-de-navegacion.html

[17] Prug. Desarrollo de Software. [En línea]. [Consultado: 2008/02/26]. Disponible en: http://www.prug.solucionesracionales.com/node/12

[18] Rational Software. Rational Unified Process Best Practices for Software Development Teams, 1998 [En línea] [Consultado: 2007/11/22]. Disponible en: http://www.rational.com

[19] Scribd. RUP. [En línea]. [Consultado: 2008/02/26]. Disponible en: http://www.scribd.com/doc/297224/RUP

[20] G. Solenzal, S. Díaz. Trabajo de Diploma ¨ Multimedia Auto_Aprende¨. Instituto Superior Politécnico ¨ José Antonio Echeverría¨, Cuidad de la Habana, 2006.

[21] Tramullas. Mapas de Navegación. [En línea]. [Consultado: 2008/02/27]. Disponible en: http://www.tramullas.com/ai/mapas-06-00.pdf

[22] U. Naranjo. Definiendo Arquitectura. [En línea]. [Consultado: 2008/02/20]. Disponible en: http://www.lucasian.com/idocs/LucasianLabs.UJAVERIANA.MNARANJO.DEF-ARQUITECTURA.ver1.6.0.ppt

[23] U. Valle. Análisis del Negocio. [En línea]. [Consultado: 2008/02/27]. Disponible en: http://www.eisc.univalle.edu.co/materias/Material_Desarrollo_Software/AnalisisNegocio.pdf

[24] Vico. Introducción a RUP. [En línea]. [Consultado: 2007/11/12]. Disponible en: http://www.vico.org/doc

[25] Willydev. Informaciones sobre RUP. [En línea]. [Consultado: 2007/11/09]. Disponible en: http://www.willydev.net/descargas/articulos/general/cualxpfddrup.PDF

[26] Wikipedia. Proceso Unificado del Racional. [En línea]. [Consultado: 2007/10/10]. Disponible en: http://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational

[27] Wikipedia. RUP. [En línea]. [Consultado: 2007/12/03]. Disponible en: http://es.wikipedia.org/wiki/RUP

[28] Y. Gómez. Metodologías para el Desarrollo de Aplicaciones Web. [En línea]. [Consultado: 2008/02/20]. Disponible en: http://www.eici.ucm.cl/Academicos/ygomez/descargas/Ing_Sw2/apuntes/DASBD-Metodolog-ADasParaElDesarrolloDeaplicacionesWeb_UWE.pdf

Glosario de Términos

A

Actividades: Es una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos.

Actor: Alguien o algo, fuera del sistema o negocio que interactúa con él.

Análisis (Flujo de Trabajo): Flujo de trabajo fundamental cuyo propósito principal es analizar los requisitos descritos en la captura de requisitos, mediante su refinamiento y estructuración.

Artefactos: Productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.

Arquitectura: Es el esqueleto o base de una aplicación, en esta se analiza la aplicación desde varios puntos de vista.

B

Base de Datos (BD): Conjunto de datos interrelacionados, almacenados con carácter más o menos permanente en la computadora, puede ser considerada una colección de datos variables en el tiempo.

C

Casos de Uso: Representa un proceso dentro del negocio que se estudia, por lo que se corresponde con una secuencia de acciones con un orden lógico y que producen un resultado observable para ciertos actores del negocio.

Clase: Una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica.

Ciclo de vida del software: Ciclo que cubre cuatro fases en el siguiente orden: inicio, elaboración, construcción y transición.

Clases de Análisis: Se centran en los requisitos funcionales y son evidentes en el dominio del problema porque representan conceptos y relaciones del dominio. Tienen atributos y entre ellas se establecen relaciones de asociación, agregación / composición, generalización / especialización y tipos asociativos.

Cliente: Persona, organización o grupo de personas que encarga la construcción de un sistema, ya sea empezando desde cero, o mediante el refinamiento de versiones sucesivas.

Componente: Se usa para modelar los elementos físicos que puedan hallarse en un nodo; por lo que empaquetan elementos como clases, colaboraciones e interfaces.

D

Desarrollador: Trabajador participante en un flujo de trabajo fundamental. Por ejemplo, un ingeniero de casos de uso, un ingeniero de componentes, etc.

Diagrama: La representación grafica de un conjunto de elementos, usualmente representado como un grafo conectado de vértices (elementos) y arcos (relaciones).

Diagrama de Actividad: Un diagrama que muestra el flujo de actividad a actividad; los diagramas de actividad tratan la vista dinámica de un sistema. Un caso especial de diagrama de estados en el cual todos o casi todos los estados son estados de acción y en el cual todas o casi todas las transiciones son disparadas por la terminación de las acciones en los estados de origen.

Diagrama de Caso de Uso: Un diagrama que muestra un conjunto de casos de uso y de actores y sus relaciones; los diagramas de casos de uso muestran los casos de uso de un sistema desde un punto de vista estático.

Diagrama de Clases: Un diagrama que muestra un conjunto de clases, interfaces y colaboraciones y las relaciones entre estos; los diagramas de clases muestran el diseño de un sistema desde el punto de vista estático; un diagrama que muestra una colección de elementos (estáticos) declarativos.

Diagrama de Colaboración: Un diagrama de interacción que enfatiza la organización estructural de los objetos que envían y reciben mensajes; un diagrama que muestra las interacciones organizadas alrededor de instancias y de los enlaces entre ellas.

Diagrama de Componentes: Un diagrama que muestra un conjunto de componentes y sus relaciones; los diagramas de componentes muestran los componentes de un sistema desde un punto de vista estático.

Diagrama de Despliegue: Un diagrama que muestra un conjunto de nodos y sus relaciones; los diagramas de estados tratan la vista dinámica de un sistema.

Diagrama de Interacción: Un diagrama que muestra una interacción, consistente en un conjunto de objetos y sus relaciones, incluyendo los mensajes que pueden ser enviados entre ellos; los diagramas de interacción tratan la vista dinámica de un sistema; un término genérico que se aplica a varios tipos de diagramas que enfatizan las interacciones de objetos, incluyendo diagramas de colaboración, diagramas de secuencia y diagramas de actividad.

Diagrama de Objetos: Un diagrama que muestra un conjunto de objetos y sus relaciones en un momento determinado; los diagramas de objetos muestran el diseño o los procesos desde el punto de vista estático.

Diagrama de Secuencia: Un diagrama de interacción que hace énfasis en la ordenación temporal de los mensajes.

Diseño (Flujo de Trabajo): Flujo de trabajo fundamental cuyo propósito principal es el de formular modelos que se centran en los requisitos no funcionales y el dominio de la solución, y que prepara la implementación y pruebas del sistema.

F

Framework: Véase marco de trabajo.

G

GRASP: Es un acrónimo que significa General Responsibility Asignment Software Patterns (patrones generales de software para asignar responsabilidades)

Glosario de Términos: Términos comunes que se utilizan para describir el sistema.

I

Interfaz: Son utilizadas en el modelo de implementación para especificar las operaciones implementadas por componentes y subsistemas de implementación. Los componentes y subsistemas pueden tener dependencias de uso sobre interfaces.

M

Marco de trabajo: Un patrón de arquitectura que proporciona una plantilla extensible para aplicaciones dentro de un dominio específico.

Modelo de CUN: Describe los procesos de un negocio (casos de uso del negocio) y su interacción con elementos externos (actores), tales como socios y clientes.

Modelo de Objeto: Describe cómo colaboran los trabajadores y entidades del negocio dentro del flujo de trabajo del proceso del negocio.

Modelo de Análisis: Contiene clases del análisis y sus objetos organizados en paquetes que colaboran.

Modelo lógico de los datos: Proveer de una vista de las entidades lógicas de datos y sus relaciones con independencia de la plataforma de base de datos a utilizar.

N

Negocio: Cualquier ambiente o entorno en cual esta enmarcado el problema.

Nodo: Un elemento físico que existe en tiempo de ejecución y que representa un recurso computacional, que en general tiene al menos alguna memoria y a menudo capacidad de procesamiento.

O

Objeto: Una manifestación concreta de de una abstracción; una entidad sobre la que pueden aplicarse un conjunto de operaciones y que tiene un estado que almacena los efectos de las operaciones; un sinónimo de instancia.

P

Patrón: Pareja de *problema / solución* con un nombre, que codifica (estandariza) buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades.

Paquete de análisis: Organiza los artefactos del análisis en piezas manejables.

Proceso: Un flujo de control pesado que puede ejecutarse concurrentemente con otros procesos.

Proceso del Negocio: Funciones que se desarrollan en el ambiente o entorno que definimos como negocio.

Prototipo de interfaz de usuario: Fundamentalmente, un prototipo ejecutable de una interfaz de usuario, pero que puede, en momentos iniciales del desarrollo, consistir únicamente en dibujos en papel, diseños de plantillas, etc.

Proyecto: Esfuerzo de desarrollo para llevar un sistema a lo largo de un ciclo de vida.

Pruebas: Las pruebas es una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones o requerimientos específicos, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente.

R

Requisitos (flujo de trabajo): Flujo de trabajo fundamental cuyo propósito principal es orientar el desarrollo el desarrollo hacia el sistema correcto. Esto se lleva a cabo mediante la descripción de los requisitos del sistema de forma tal que se pueda llegar a un acuerdo con el cliente (incluyendo los usuarios) y los desarrolladores del sistema, acerca de lo que el sistema debe hacer o no.

Rol: Papel que desempeña cada integrante dentro del ciclo de vida de un proyecto de software.

S

Sistema: Una colección de subsistemas organizados para llevar a cabo un propósito específico y descritos por un conjunto de modelos, posiblemente desde distintos puntos de vista.

Subsistema: Una agrupación de elementos, de los que algunos constituyen una especificación del comportamiento ofrecido por los otros elementos contenidos.

Stakeholders: La definición más correcta es parte interesada, es decir, cualquier persona o entidad que es afectada por las actividades de una organización; por ejemplo, los trabajadores de esa organización, sus accionistas, las asociaciones de vecinos, sindicatos, clientes, organizaciones civiles y gubernamentales, etc.

T

Trabajadores: Representa a personas, o sistemas (software) dentro del negocio que son las que realizan las actividades que están comprendidas dentro de un caso de uso; en un futuro se convertirán en usuarios del sistema que se quiere construir.

U

UML: "Unified Modeling Language" Lenguaje gráfico que brinda un vocabulario y reglas para especificar, construir, visualizar y documentar los artefactos de un sistema utilizando el enfoque orientado a objetos.

Usuario: Persona que interactúa con un sistema.

Anexo 1

ENCUESTA A DESARROLLADORES DE SOFTWARE

2007-2008

Institución _____

La información que le solicitamos a continuación, acerca de RUP como metodología utilizada para el desarrollo de software por usted es anónima.

1. ¿Utilizas RUP como metodología para el desarrollo de software?

□ Si

□ No

2. ¿Utiliza RUP para proyectos?

a. A corto plazo _____

b. A largo plazo _____

c. Ambos _____

3. ¿En qué tipo de aplicaciones considera usted que la metodología RUP es la más adecuada?

□ Multimedia

□ Aplicaciones de escritorio

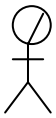
□ Aplicaciones WEB

4. ¿Consideras a RUP como una metodología eficiente para el desarrollo de proyectos con grupos pequeños de trabajo?

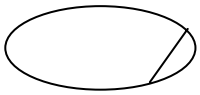
5. Sugerencias fundamentales que usted propone de metodología RUP para cada tipo de aplicación.

Anexo 2

Estereotipos usados en el Modelo de Casos de Uso del Negocio



Actor del

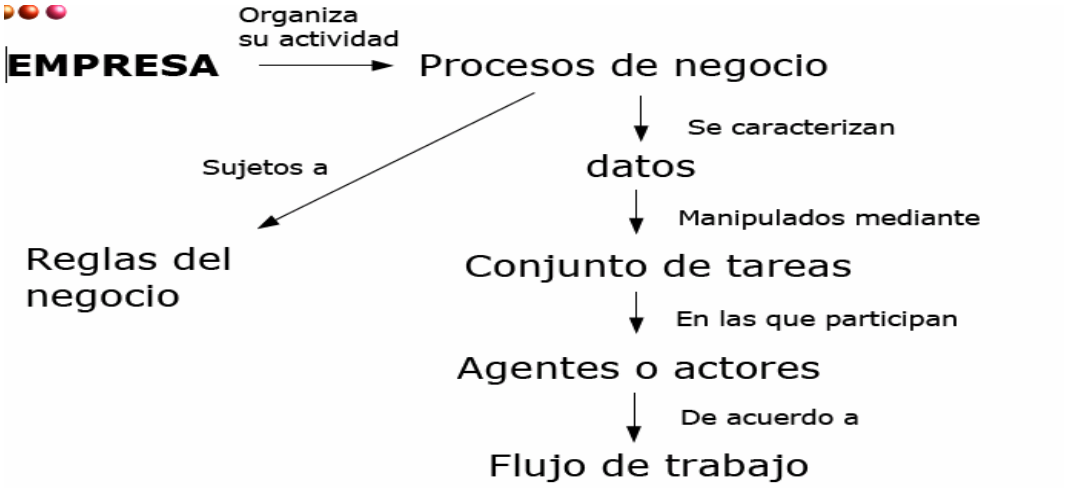


Caso de Uso

Anexo 3

Descripción del Negocio [11]

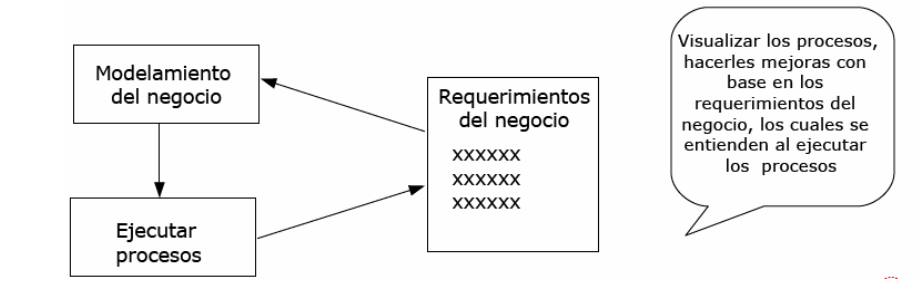
Se hace una descripción de todos los proceso en el negocio:



Modelamiento del Negocio

- Entender la estructura y dinámica de la organización para la cual se desarrollará el S.I.
- Identificar y entender los problemas al interior de la organización e identificar posibles soluciones.
- Asegurar que los clientes, usuarios y desarrolladores tienen entendimiento común de la organización.

El modelamiento del negocio se puede ver como un simple escenario donde se entienda o se contrasten los procesos del negocio.



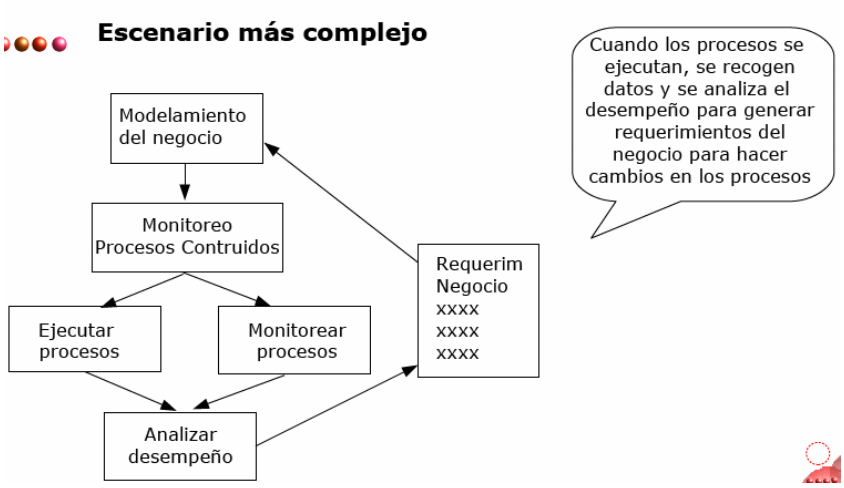


Diagrama Actividad del Proceso de Modelamiento del Negocio

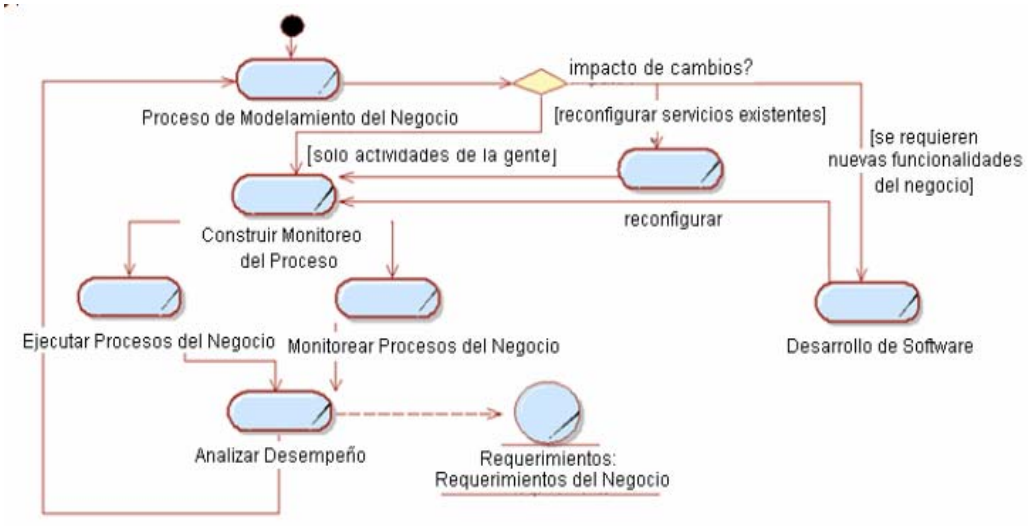
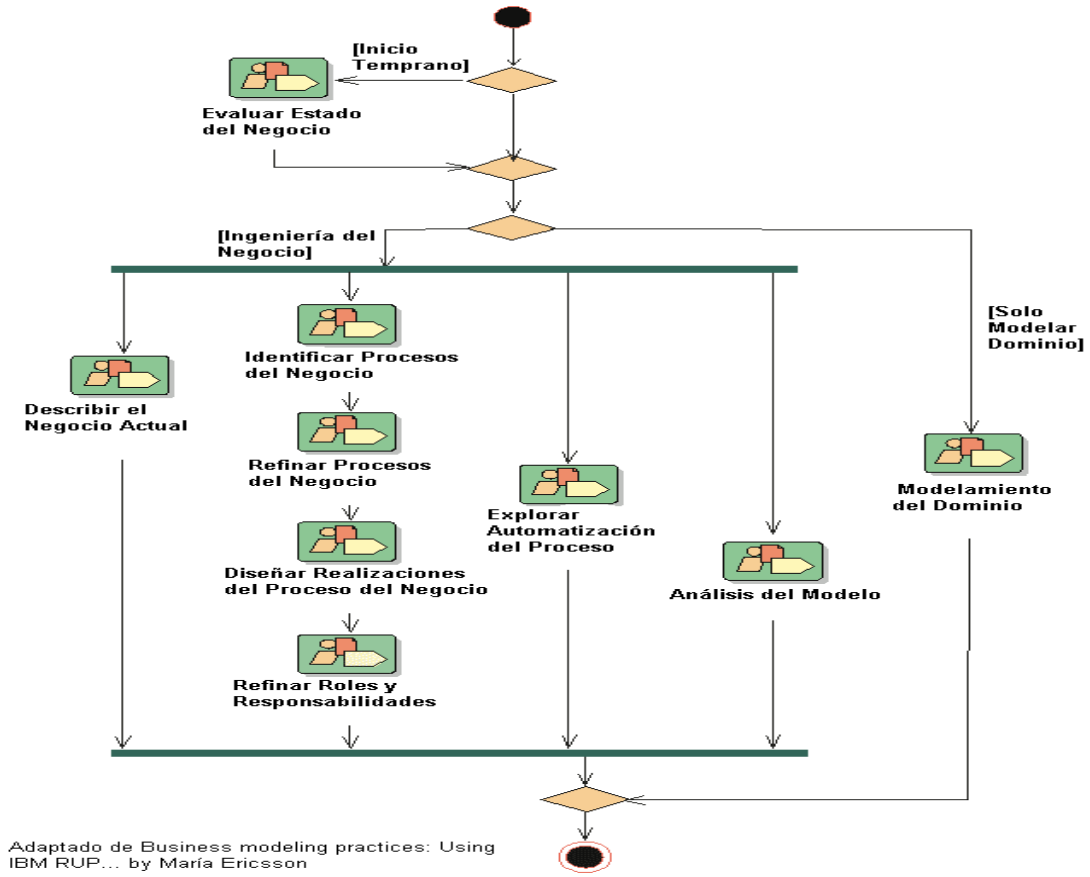


Diagrama del Proceso de Modelamiento del Negocio



Actividades del Modelamiento del Negocio

- Describir el negocio actual (diagrama de contexto)
- Identificar los procesos y actores de la organización bajo estudio: actores internos y externos (diagrama de casos de uso del negocio)
- Refinar definiciones de los procesos del negocio (descripción de casos de uso)
- Diseñar las realizaciones de los procesos del negocio (diagramas de secuencia/actividad/flujo de trabajo)

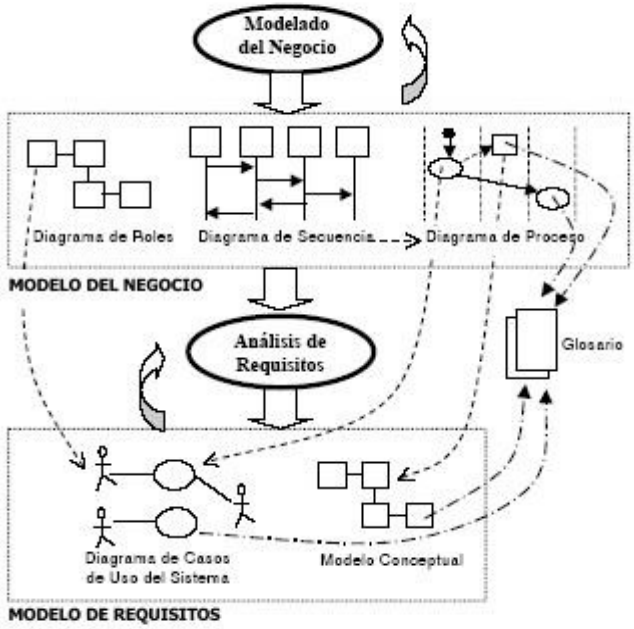


Fig. 1. Relaciones de trazabilidad entre los modelos de negocio y de requisitos

Anexo 4

Los criterios para agrupar CUN podrían ser los siguientes:

- Casos de uso de negocio que se ocupan de la misma información.
- Casos de uso de negocio usados por el mismo grupo de actores.
- Casos de uso de negocio que se ejecutan a menudo en una sucesión.
- Los casos de uso de negocio más importantes.
- Un caso de uso de negocio específico y sus relaciones con los actores de negocio y otros casos de uso de negocio.

Anexo 5

La descripción textual de un caso de uso de negocio se formaliza en un documento generalmente llamado "Especificación del caso de uso de negocio". Este documento puede tener el siguiente formato:

Nombre del caso de uso del negocio:	Nombre
Actores del negocio:	Lista de actores que se relacionan con el caso de uso, indicando quién lo inicia.

<table>
<tr><td>Propósito:</td><td colspan="2">Breve descripción del objetivo del proceso.</td></tr>
<tr><td colspan="3">Resumen:
Descripción del proceso completo indicando quién inicia y cómo se inicia, cuál es el flujo de trabajo a grandes rasgos y quién finaliza el proceso y cómo se hace. La descripción debe mencionar a los actores y trabajadores del negocio y a las actividades más importantes que se ejecutan.</td></tr>
<tr><td>Casos de uso asociados:</td><td colspan="2">Listado de casos de uso incluidos y extendidos de este caso de uso base, indicando el tipo de relación.</td></tr>
<tr><td colspan="3">Flujo de trabajo</td></tr>
<tr><td>Acción del actor</td><td colspan="2">Respuesta del negocio</td></tr>
<tr><td>Se indica al actor (o actores) y la interacción que tiene con el negocio.</td><td colspan="2">Se describe el flujo de trabajo indicando todas las actividades del negocio que ocurren en el orden que se suceden, cuál es el trabajador del negocio que las realiza y su relación con las entidades del negocio. Deben quedar claros los puntos intermedios en los que puede finalizar el proceso.</td></tr>
<tr><td>Prioridad:</td><td colspan="2">Indicar cuál es la prioridad de este proceso dentro del negocio que se modela.</td></tr>
<tr><td>Mejoras:</td><td colspan="2">Mejoras que tendrá el proceso cuando algunas de sus actividades sean automatizadas.</td></tr>
<tr><td colspan="3">Cursos alternos:
Comportamiento que no está en el flujo normal y que ocurre bajo ciertas condiciones que pueden darse en el flujo normal.</td></tr>
</table>

El flujo de trabajo normal de un caso de uso de negocio puede tener un flujo básico de actividades y flujos alternativos de actividades. El flujo básico cubre lo que siempre ocurre cuando el caso de uso de negocio es ejecutado, y el flujo alternativo describe alternativas que pueden

darse cuando se produce el caso de uso, pero que no se dan de forma excepcional como ocurre con los flujos alternativos. Por ejemplo, en la descripción de la producción de una pieza, en dependencia de qué pieza se va a construir, hay un grupo de actividades diferentes porque pasan por diferentes máquinas y la actividad es realizada por trabajadores diferentes.

Anexo 6

Algunas consideraciones en la creación de Diagramas de Actividades

Los Diagramas de Actividades permiten muchas libertades, lo que a veces estimula a los creadores a incluir un alto nivel de detalle. En definitiva, un modelo de comunicación requiere un adecuado nivel de detalle para ubicar el problema a resolver. La claridad y brevedad son dos atributos importantes para evitar la sobrecarga y limitarse sólo a presentar los aspectos claves de los flujos de los casos de uso.

Se sugieren seguir las siguientes reglas, entre otras;

- No intentar mostrar elementos de diseño. Centrarse en las necesidades del cliente y no moverse hacia el espacio de la solución, es decir, seguir el principio de enfocarse a la funcionalidad, desde la perspectiva del usuario. Por ejemplo, crear una actividad que sea Conectar a la Base de Datos Oracle, estaría violando ese principio.
- No sustituir los diagramas de actividad por la descripción de los casos de uso.
- Limitar el nivel de complejidad de cada diagrama. Para ello:
- Si hay más de 3 posibles caminos (alternos o de excepción), usar diagramas adicionales para mejorar la comprensión.
- Usar Swimlanes (calles) para separar responsabilidades.
- En la medida de lo posible utilizar un diagrama por cada caso de uso.
- Mantener los modelos. Los diagramas deben actualizarse cuando se modifiquen los casos de uso

Anexo 7

Clasificación de las Reglas del Negocio

1. Reglas de estructura

- Término: Conceptos en el contexto del negocio.

Ejemplo: Estudiante, Libro

- Modelo de datos: Controla que la información básica almacenada para cada atributo o propiedad de un concepto es válida.

Ejemplo: La cantidad de libros de un mismo título es mayor que cero

- Relación: Controla las relaciones entre los datos.

Ejemplo: El estudiante solicita un libro. A la asociación entre el libro y el estudiante se le denomina Préstamo.

2. Reglas de derivación

- Inferencia: Especifican que un hecho es cierto por inferencia.

Ejemplo: Un estudiante que debe un préstamo de un libro se convierte en un estudiante moroso.

- Cálculo: Controla la obtención de información que se puede calcular a partir de la ya existente.

Ejemplo: La cantidad de libros que un estudiante tiene en préstamo es la suma de los préstamos en los que está involucrado.

3. Reglas de acción

- Flujo: Determinan y limitan cómo fluye la información a través de un sistema.

Ejemplo: Un estudiante solicita un libro en préstamo en una biblioteca, el personal que lo atiende verifica si es un estudiante moroso. En caso negativo, el personal procede a registrar el préstamo y le entrega el libro. Si no es posible, se le informa y recoge el libro para colocarlo en su estante.

- Restricciones de operaciones: Especifican condiciones que deben ser ciertas para asegurarse que una operación se ejecute correctamente.

Ejemplo: A un estudiante no se le puede prestar otro libro si está clasificado como moroso.

- Estímulo y respuesta: Restringen el comportamiento especificando cuándo y qué condiciones deben cumplirse para que una operación de respuesta sea inmediatamente ejecutada.

Ejemplo: Si se ha llegado a la fecha en que un estudiante debe entregar un libro y no lo ha hecho, se procede a enviarle un e-mail.

Anexo 8

Una CRC es la abreviatura de CLASES -RESPONSABILIDADES- COLABORACION. Es una técnica para identificar responsabilidades. y Su objetivo es facilitar la comunicación y documentar los resultados del Análisis.

Para cada clase identificada se rellenará una tarjeta de este tipo y se especificará su finalidad así como otras clases con las que interactúe. Esta información se pasa directamente a un diagrama de clase; las responsabilidades coinciden con los métodos de clase, las colaboraciones se traducen en asociaciones entre clases.

Ejemplo

Nombre de Clase: Documento_Ingreso
Superclase

Subclase	
Responsabilidad	Colaboración
Mantener los datos del Documento de Ingreso.	Gestor CC_BuscaDocIngreso Gestor CC_RegistraDocIngreso

Anexo 9

Patrón Layers

Es un patrón arquitectónico que ayuda a estructurar aplicaciones que pueden descomponerse en grupos de subtareas de forma que las tareas de cada grupo se encuentran en el mismo nivel de abstracción.

Anexo 10

Implicaciones de los conceptos OO para el diseño de casos de prueba

La clase OO es el objetivo para el diseño de casos de prueba. La ejecución de pruebas necesita de informes sobre el estado concreto y abstracto de un objeto.

El encapsulamiento es un concepto esencial de diseño para la OO, éste puede crear un obstáculo para la ejecución de la prueba, como hacer difícil la obtención de información sobre el estado concreto y abstracto de un objeto. A menos que se provea de operaciones incorporadas.

La herencia también conduce a cambios adicionales al diseñador de casos de prueba. Ya hemos notado que cada contexto de nuevo uso necesita la repetición de la prueba, incluso cuando se haya reutilizado.

En caso de que se usen subclases, instanciadas de una superclase, dentro del mismo dominio del problema, es similar a que el conjunto de casos de prueba, derivados de la clase en un contexto por completo diferente, pueda usarse al probar la subclase. Sin embargo, si se usa la subclase en un contexto diferente, los casos de prueba de la superclase tendrán poca aplicabilidad y se deberá diseñar un nuevo conjunto de pruebas.