



PROYECTO FINAL TC1/TD

- GUÍA DE CONTENIDOS -

Versión 0.9

Enero de 2006

ÍNDICE

1. INTRODUCCIÓN	6
2. REQUERIMIENTOS MÍNIMOS.....	7
2.1. CARPETA DE PROYECTO	7
3. EL PROCESO UNIFICADO DE DESARROLLO	9
3.1. CARACTERÍSTICAS ESENCIALES	9
3.1.1. <i>Proceso dirigido por Casos de Uso</i>	9
3.1.2. <i>Proceso centrado en la arquitectura</i>	10
3.1.3. <i>Proceso iterativo e incremental</i>	12
3.2. PRINCIPIOS ESENCIALES	14
3.3. VISIÓN GENERAL DE FASES, OBJETIVOS E HITOS.....	16
3.4. MAPA CONCEPTUAL DEL PROCESO UNIFICADO	18
3.5. DETALLE DE FASES, OBJETIVOS E HITOS	19
3.5.1. <i>Fase de Inicio</i>	19
3.5.1.1. Objetivo	19
3.5.1.2. Hito	19
3.5.1.3. Criterio de Evaluación.....	19
3.5.2. <i>Fase de Elaboración</i>	20
3.5.2.1. Objetivo	20
3.5.2.2. Hito	20
3.5.2.3. Criterio de Evaluación.....	20
3.5.3. <i>Fase de Construcción</i>	21
3.5.3.1. Objetivo	21
3.5.3.2. Hito	21
3.5.3.3. Criterio de Evaluación.....	21
3.5.4. <i>Fase de Transición</i>	22
3.5.4.1. Objetivo	22
3.5.4.2. Hito	22
3.5.4.3. Criterio de Evaluación.....	22
3.5.5. <i>Grado de Finalización de los Artefactos según la Fase</i>	23
3.6. ROLES. EL QUIÉN.	24
3.7. ACTIVIDADES. EL CÓMO.....	26
3.8. ARTEFACTOS. EL QUÉ.	27
3.9. DISCIPLINAS. EL CUÁNDO.	28
3.9.1. <i>Modelado del Negocio</i>	28
3.9.1.1. Objetivo	28
3.9.2. <i>Gestión de Requerimientos</i>	28
3.9.2.1. Objetivo	28
3.9.3. <i>Análisis & Diseño</i>	28
3.9.3.1. Objetivo	29
3.9.4. <i>Implementación</i>	29
3.9.4.1. Actividades.....	29
3.9.5. <i>Pruebas</i>	29
3.9.5.1. Objetivo	30
3.9.6. <i>Despliegue</i>	30
3.9.6.1. Actividades.....	30
3.9.7. <i>Gestión de Proyecto</i>	30
3.9.8. <i>Gestión de Cambios (Change management)</i>	30
3.9.9. <i>Entorno</i>	31
4. ARTEFACTOS ENTREGABLES.....	32
4.1. DESCRIPCIÓN DE LA EMPRESA.....	32
4.2. PLAN DE DESARROLLO DEL SOFTWARE	32
4.3. VISIÓN.....	32
4.4. MODELO DE CASOS DE USO DEL NEGOCIO	32
4.5. MODELO DE DOMINIO / OBJETOS DEL NEGOCIO.....	33

PROYECTO FINAL TC1/TD

4.6.	ESTUDIO DE VIABILIDAD.....	33
4.6.1.	<i>Viabilidad Legal.....</i>	33
4.6.2.	<i>Viabilidad Técnica.....</i>	33
4.6.3.	<i>Viabilidad Operativa.....</i>	34
4.6.4.	<i>Viabilidad Económico/Financiera.....</i>	35
4.6.4.1.	<i>Valor Actual Neto (VAN).....</i>	35
4.6.4.2.	<i>Costo Neto (CT)</i>	36
4.6.4.3.	<i>Valor Actual Neto (VAN).....</i>	36
4.6.4.4.	<i>Tasa Interna de Retorno (TIR) y Rendimiento del proyecto</i>	37
4.6.5.	<i>Viabilidad de Gestión</i>	38
4.6.6.	<i>Viabilidad Comercial.....</i>	38
4.7.	GLOSARIO.....	39
4.8.	MODELO DE CASOS DE USO	39
4.9.	ESPECIFICACIONES DE CASOS DE USO.	39
4.10.	ESPECIFICACIONES ADICIONALES.....	39
4.11.	MAPA DE NAVEGACIÓN	39
4.12.	PROTOTIPOS DE INTERFACES DE USUARIO	40
4.13.	DIAGRAMA DE CLASES	40
4.14.	DIAGRAMA DE SECUENCIA.....	45
4.15.	DIAGRAMA DE COMPONENTES.....	46
4.16.	DER	47
4.17.	MODELO DE DESPLIEGUE.....	51
4.18.	ESPECIFICACIÓN DE CASOS DE PRUEBA.....	53
4.19.	SOLICITUD DE CAMBIO	53
4.20.	PLAN DE ITERACIÓN	53
4.21.	EVALUACIÓN DE ITERACIÓN	53
4.22.	LISTA DE RIESGOS.....	53
4.23.	MANUAL DE INSTALACIÓN.....	54
4.24.	MATERIAL DE APOYO AL USUARIO FINAL.....	54
4.25.	PRODUCTO.....	54
4.25.1.	<i>Solución del negocio.....</i>	54
4.25.2.	<i>Dígitos verificadores verticales y horizontales.....</i>	54
4.25.3.	<i>Log-in /log-out de usuarios</i>	55
4.25.3.1.	<i>Características del acceso al sistema que se deben incluir en la carpeta de proyecto</i> 55	
4.25.4.	<i>Asignación de perfiles utilizando el modelo de Usuario/Familia/Patente.....</i>	55
4.25.5.	<i>Registración de actividades & Bitácora de accesos.....</i>	56
4.25.6.	<i>Multi-idioma</i>	56
4.25.7.	<i>Arquitectura Escalable & Extensible. Patrones de diseño</i>	56
4.25.8.	<i>Pruebas unitarias.....</i>	57
4.25.9.	<i>Indicadores de Rendimiento del Software.....</i>	57
4.25.10.	<i>Manejo de Excepciones</i>	57
4.26.	RESUMEN DE ENTREGABLES A DESARROLLAR POR CÁTEDRA.....	59
4.27.	MATRIZ DE TRAZABILIDAD ENTRE ENTREGABLES Y PROYECTO MODELO.....	60
5.	PROYECTO MODELO.....	61
5.1.	ÍNDICE	62
5.2.	GESTIÓN DEL PROYECTO	63
5.2.1.	<i>Plan de Desarrollo de Software.....</i>	63
5.2.2.	<i>Estudio de Viabilidad.....</i>	63
5.2.2.1.	<i>Viabilidad Legal</i>	63
5.2.2.2.	<i>Viabilidad Técnica</i>	63
5.2.2.3.	<i>Viabilidad Operativa</i>	64
5.2.2.4.	<i>Viabilidad Económico/Financiera.....</i>	64
5.2.2.5.	<i>Viabilidad de Gestión</i>	64
5.2.2.6.	<i>Viabilidad Comercial.....</i>	65
5.2.3.	<i>Lista de Riesgos.....</i>	65
5.2.4.	<i>Planificación del Proyecto.....</i>	65
5.2.4.1.	<i>Fase de Inicio</i>	65

PROYECTO FINAL TC1/TD

Plan de Iteración. Iteración de Inicio #I1.....	65
Evaluación de Iteración de Inicio #I1.....	65
5.2.4.2. Fase de Elaboración.....	65
Plan de Iteración. Iteración de Elaboración #E1.....	65
Evaluación de Iteración de Elaboración #E1	65
Plan de Iteración. Iteración de Elaboración #E2.....	65
Evaluación de Iteración de Elaboración #E2	65
Plan de Iteración. Iteración de Elaboración #E3.....	65
Evaluación de Iteración de Elaboración #E3	65
5.2.4.3. Fase de Construcción	65
Plan de Iteración. Iteración de Construcción #C1	65
Evaluación de Iteración de Construcción #C1	65
Plan de Iteración. Iteración de Construcción #C2	65
Evaluación de Iteración de Construcción #C2	65
Plan de Iteración. Iteración de Construcción #Cn	66
Evaluación de Iteración de Construcción #Cn	66
5.2.4.4. Fase de Transición	66
Plan de Iteración. Iteración de Transición #T1	66
Evaluación de Iteración de Transición #T1.....	66
Plan de Iteración. Iteración de Transición #T2	66
Evaluación de Iteración de Transición #T2.....	66
5.3. MODELADO DEL NEGOCIO	67
5.3.1. <i>Descripción de la empresa</i>	67
5.3.2. <i>Modelado del Negocio</i>	67
5.3.2.1. Modelo de Casos de Uso del Negocio	67
5.3.2.2. Modelo del Dominio	68
5.3.2.3. Modelo de Objetos del Negocio	68
5.3.2.3.1. Modelo de Objetos de Vender Productos	69
5.3.2.3.2. Modelo de Objetos de Seguimiento y Consulta de Productos	69
5.3.2.3.3. Modelo de Objetos de Reponer Stock.....	69
5.3.2.3.4. Modelo de Objetos de Modificar Catálogo.....	70
5.3.2.3.5. Modelo de Objetos de Realizar Entrega.....	70
5.4. REQUISITOS	71
5.4.1. <u>Documento Visión</u>	71
5.4.2. <u>Documento Glosario</u>	71
5.4.3. <i>Solicitud de Cambio</i>	71
5.4.4. <i>Modelo de Casos de Uso</i>	71
5.4.4.1. Gestión de Ventas.....	71
5.4.4.2. Gestión de Almacén	73
5.4.4.3. Gestión de Envíos.....	74
5.4.4.4. Departamento de Logística.....	75
5.4.4.5. Departamento de Marketing.....	76
5.4.4.6. Departamento de Contabilidad/Facturación.....	77
5.4.4.7. Departamento de Recursos Humanos.....	78
5.4.5. <i>Especificación de Casos de Uso</i>	79
5.4.5.1. Gestión de Ventas.....	79
5.4.5.1.1. <u>Especificación del caso de uso "control de estadísticas"</u>	79
5.4.5.1.2. <u>Especificación del caso de uso "consultar catálogo"</u>	79
5.4.5.1.3. <u>Especificación del caso de uso "otorgar incentivos"</u>	79
5.4.5.1.4. <u>Especificación del caso de uso "elaborar pedido"</u>	79
5.4.5.1.5. <u>Especificación del caso de uso "elaborar pedido online"</u>	79
5.4.5.1.6. <u>Especificación del caso de uso "gestión de clientes"</u>	79
5.4.5.2. Gestión de Almacén	79
5.4.5.2.1. <u>Especificación del caso de uso "consultar pedidos no atendidos"</u>	79
5.4.5.2.2. <u>Especificación del caso de uso "atender pedido"</u>	79
5.4.5.2.3. <u>Especificación del caso de uso "cancelar pedido atendido"</u>	79
5.4.5.2.4. <u>Especificación del caso de uso "incidencia pedido"</u>	79
5.4.5.2.5. <u>Especificación del caso de uso "pasar pedido a envío"</u>	79
5.4.5.2.6. <u>Especificación del caso de uso "reposición de stock"</u>	79
5.4.5.3. Gestión de Envíos.....	79
5.4.5.3.1. <u>Especificación del caso de uso "consultar pedidos a enviar"</u>	79
5.4.5.3.2. <u>Especificación del caso de uso "introducir recibos"</u>	79
5.4.5.3.3. <u>Especificación del caso de uso "realizar envío"</u>	79

PROYECTO FINAL TC1/TD

5.4.5.4.	Departamento de Logística.....	79
5.4.5.4.1.	Especificación del caso de uso "compra a proveedores"	79
5.4.5.4.2.	Especificación del caso de uso "gestión de regiones"	79
5.4.5.4.3.	Especificación del caso de uso "reabastecer almacén"	79
5.4.5.5.	Departamento de Marketing.....	79
5.4.5.5.1.	Especificación del caso de uso "confeccionar catálogo"	79
5.4.5.5.2.	Especificación del caso de uso "política de ventas"	79
5.4.5.5.3.	Especificación del caso de uso "realizar oferta"	79
5.4.5.6.	Departamento de Contabilidad/Facturación.....	79
5.4.5.6.1.	Especificación del caso de uso "cobro a clientes"	79
5.4.5.7.	Departamento de Recursos Humanos.....	79
5.4.5.7.1.	Especificación del caso de uso "entrevista de trabajo"	79
5.4.5.7.2.	Especificación del caso de uso "gestión de nóminas"	79
5.4.5.7.3.	Especificación del caso de uso "gestión de personal"	79
5.4.5.7.4.	Especificación del caso de uso "redistribución de personal"	79
5.4.6.	Especificaciones Adicionales	80
5.4.7.	Especificación de Casos de Prueba	80
5.4.7.1.	Base de datos.....	80
5.4.7.1.1.	Especificación de la Base de Datos de Prueba	80
5.4.7.2.	Gestión de almacén	80
5.4.7.2.1.	Especificación de Caso de Prueba "Consultar pedidos no atendidos"	80
5.4.7.2.2.	Especificación de Caso de Prueba "Atender Pedido"	80
5.4.7.2.3.	Especificación de Caso de Prueba "Cancelar Pedido Atendido"	80
5.4.7.2.4.	Especificación de Caso de Prueba "Incidencia Pedido"	80
5.4.7.2.5.	Especificación de Caso de Prueba "Pasar Pedido a Envío"	80
5.4.7.3.	Gestión de Ventas.....	80
5.4.7.3.1.	Especificación de Caso de Prueba "Elaborar Pedido"	80
5.5.	ANÁLISIS/DISEÑO	81
5.5.1.	<i>Diagrama de Clases</i>	81
5.5.2.	<i>Diagramas de Secuencia</i>	82
5.5.3.	<i>Diagramas de Transición-Estados</i>	83
5.5.4.	<i>Diagramas de Actividad</i>	84
5.5.5.	<i>DER</i>	85
5.6.	IMPLEMENTACIÓN	86
5.6.1.	<i>Mapa de Navegación</i>	86
5.6.2.	<i>Prototipo de Interfaces de Usuario</i>	86
5.6.2.1.	Interfaces Comunes.....	86
5.6.2.2.	Gestión de Ventas.....	88
5.6.2.3.	Gestión de Almacén	90
5.6.3.	<i>Diagrama de Componentes</i>	96
5.6.3.1.	Diagrama Global de Paquetes	96
5.6.3.2.	Diagrama de Componentes Comunes	97
5.6.3.3.	Diagrama de Componentes de Almacén.....	98
5.6.3.4.	Diagrama de Componentes de Ventas.....	99
5.6.4.	<i>Diagrama de Despliegue</i>	100
6.	PLANTILLAS DE ARTEFACTOS DEL PROCESO UNIFICADO	101
7.	REFERENCIAS Y LECTURA RECOMENDADA	102

1. Introducción

La materia Trabajo de Campo I tiene una carga horaria curricular de 96 horas cuatrimestrales y además requiere, dependiendo de la experiencia del alumno en este tipo de proyectos, de 200 horas cuatrimestrales extracurriculares para la elaboración de la carpeta.

El cuatrimestre se divide en 16 semanas, tiempo durante el cual se deberá completar la presentación de la carpeta de proyecto de acuerdo a las pautas establecidas por el docente.

En la presente guía se detallan los conceptos necesarios para la confección de la carpeta de proyecto correspondiente a la materia Trabajo de Campo I.

Dentro de ella encontrarán detallados los distintos puntos que deben ser incluidos en la carpeta, conceptos teóricos de algunos de los temas, las nomenclaturas a utilizar en cada una de las fases, ejemplos de cada una de las partes a desarrollar y el cronograma de las entregas parciales que deberán realizar los alumnos. Es importante destacar que para la confección de esta carpeta, el proyecto deberá planificarse y desarrollarse de acuerdo al paradigma de proceso de desarrollo establecido por el [Proceso Unificado](#) (EUP – Enterprise Unified Process).

Es importante, por parte del alumno, respetar las fechas establecidas para las presentaciones de los avances de la carpeta, para evitar complicaciones sobre e final de la cursada provocada por errores de fases iniciales que tienen incidencia en las fases finales.

En la sección Entregables se encuentra especificado en qué materia de la carrera se deberá el alumno desarrollar el material indicado. Se indica además si el mismo es de carácter obligatorio para la aprobación del proyecto o si es opcional y deberá confeccionarse de acuerdo al criterio del docente y/o alumno.

Cátedra	Carácter
ASA	Requerido
ASA	Opcional
TC1	Requerido
TC1	Opcional
TD	Requerido
TD	Opcional

2. Requerimientos mínimos

2.1. Carpeta de Proyecto

Deberá ser presentada en un bibliorato y confeccionada en hoja tamaño A4, impresa a una carilla, utilizando letra Arial o Verdana, tamaño 11 como máximo para el cuerpo, 12 para los subtítulos y 14 para los títulos.

En el lomo del bibliorato se deben consignar los siguientes datos:

- Materia
- Localización y curso
- Apellido y nombre del alumno
- Año de cursada

Los márgenes de la hoja deben ser:

- Izquierdo: 3 cm.
- Derecho: 2 cm.
- Superior: 4 cm.
- Inferior: 1,5 cm.

En el encabezado de todas las hojas de la carpeta debe constar:

- Nombre de la Universidad y Facultad.
- Nombre de materia (ej.: Trabajo de Campo I)
- Nombre del docente.
- Localización, comisión y turno donde cursa.
- Año de cursada (ej: 2006)
- Título o tema del trabajo.
- Nombre/s y apellido/s completos del alumno.
- Número de legajo.
- Nombre de la sección o etapa a la que pertenece.
- Número de página.
- La inclusión de un “logo” o imagen en el encabezado es optativo. (Este logo es el de su empresa y no el de la empresa que está relevando)

En el índice deben estar detallados todos los títulos y subtítulos con las páginas donde están ubicados.

Los contenidos deberán estar divididos en secciones, cada una de las cuales representa una de las fases e iteraciones del modelo de Proceso Unificado del desarrollo de software:

1. Fase de Inicio
 - 1.1. Iteración de Inicio #I1
2. Fase de Elaboración
 - 2.1. Iteración de Elaboración #E1
 - 2.2. Iteración de Elaboración #E2
 - 2.3. Iteración de Elaboración #E3
3. Fase de Construcción
 - 3.1. Iteración de Construcción #C1
 - 3.2. Iteración de Construcción #C2
 - 3.3. Iteración de Construcción #Cn
4. Fase de Transición
 - 4.1. Iteración de Transición #T1
 - 4.2. Iteración de Transición #T2
5. Lista de Riesgos

PROYECTO FINAL TC1/TD

NOTA: Estas son las fases e iteraciones típicas del Proceso Unificado, puede consultar la [sección específica](#) para obtener más información sobre este punto.

Cada fase debe dar comienzo a una nueva sección y tener una carátula con su nombre que la identifique.

Toda documentación adicional que se desee incorporar a la carpeta deberá hacerse al final de la misma como Apéndices.

- Un apéndice por documento
- Identificados por una letra (ej.: Apéndice A)

No deberá tener faltas de ortografía. La existencia de más de 10 errores ortográficos o ausencias de tildes es causa suficiente para la no aprobación de la carpeta.

3. El Proceso Unificado de Desarrollo

El Proceso Unificado de Desarrollo es un proceso de ingeniería del software, bien definido y estructurado. Presenta un marco de proceso adaptable a las necesidades y características de cada proyecto específico.

3.1. Características Esenciales

3.1.1. Proceso dirigido por Casos de Uso

Los Casos de Uso son una técnica de captura de requisitos que fuerza a pensar en términos de importancia y *creación de valor* para el usuario y no sólo en términos de funciones que sería bueno contemplar. Se define un Caso de Uso como un fragmento de funcionalidad del sistema que proporciona al usuario un valor añadido. Los Casos de Uso representan los requisitos funcionales del sistema.

En el Proceso Unificado los Casos de Uso no son sólo una herramienta para especificar los requisitos del sistema. También guían su diseño, implementación y prueba. Los Casos de Uso constituyen un elemento integrador y una guía del trabajo como se muestra en la Figura 2.



Figura 2: Los Casos de Uso integran el trabajo.

Los Casos de Uso no sólo inician el proceso de desarrollo sino que proporcionan un hilo conductor, permitiendo establecer trazabilidad entre los artefactos que son generados en las diferentes actividades del proceso de desarrollo.

Como se muestra en la Figura 3, basándose en los Casos de Uso se crean los modelos de análisis y diseño, luego la implementación que los lleva a cabo, y se verifica que efectivamente el producto implemente adecuadamente cada Caso de Uso. Todos los modelos deben estar sincronizados con el modelo de Casos de Uso.

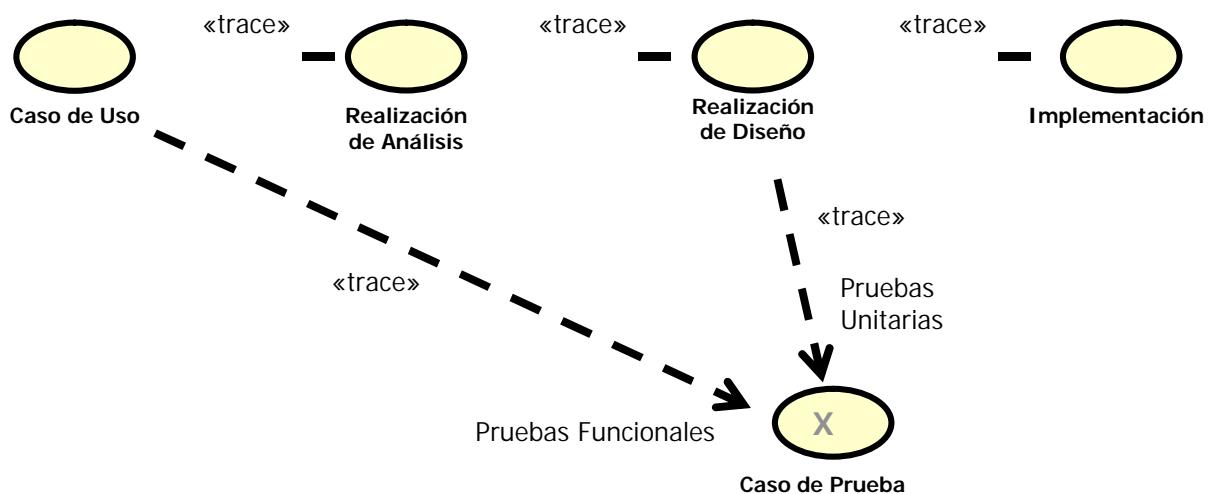


Figura 3: Trazabilidad a partir de los Casos de Uso.

3.1.2. Proceso centrado en la arquitectura

La arquitectura de un sistema es la organización de sus partes más relevantes, lo que permite tener una visión común entre todos los involucrados (desarrolladores y usuarios) y una perspectiva clara del sistema completo, necesaria para controlar el desarrollo y establecer los fundamentos para que analistas, diseñadores, programadores, etc. trabajen en una línea común que permita alcanzar los objetivos y necesidades del sistema de información.

Una arquitectura software se selecciona y diseña en base a unos objetivos y restricciones. Los **objetivos** son aquellos prefijados para el sistema de información, pero no solamente los de tipo funcional, también otros objetivos como calidad del sistema, el rendimiento, la mantenibilidad, auditabilidad, flexibilidad e interacción con otros sistemas de información. Las **restricciones** son aquellas limitaciones derivadas de las tecnologías disponibles para implementar sistemas de información.

Algunas arquitecturas son más recomendables de implementar con ciertas tecnologías mientras que otras tecnologías no son aptas para determinadas arquitecturas. Por ejemplo, no es viable emplear una arquitectura software de tres capas para implementar sistemas en tiempo real.

La arquitectura involucra los aspectos estáticos y dinámicos más significativos del sistema, está relacionada con la toma de decisiones que indican cómo tiene que ser construido el sistema y ayuda a determinar en qué orden.

La arquitectura software define, de manera abstracta, los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación entre ellos. Toda arquitectura de software debe ser implementable en una arquitectura física, que consiste simplemente en determinar qué computadora tendrá asignada cada tarea de computación. Asimismo, se ve influenciada por la plataforma software, sistema operativo, gestor de bases de datos, protocolos, consideraciones de desarrollo como sistemas heredados. Muchas de estas restricciones constituyen requisitos no funcionales del sistema.

En el caso del Proceso Unificado además de utilizar los Casos de Uso para guiar el proceso se presta especial atención al establecimiento temprano de una buena arquitectura que no se vea fuertemente impactada ante cambios posteriores durante la construcción y el mantenimiento.

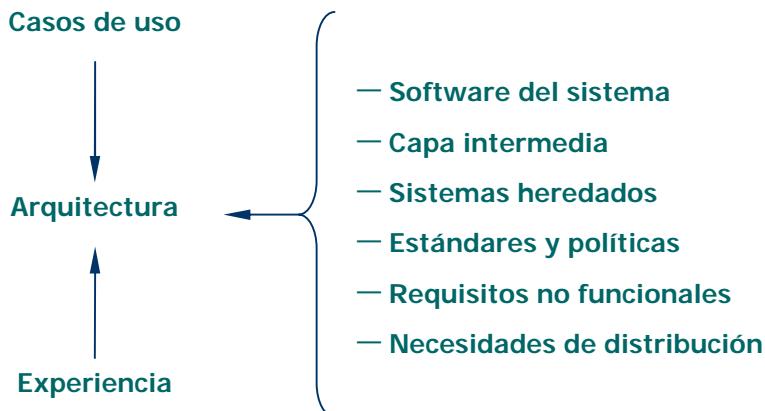


Figura 4: Componentes de la arquitectura

Cada producto tiene tanto una **función** como una **forma**. La función corresponde a la funcionalidad reflejada en los Casos de Uso y la forma la proporciona la arquitectura. Existe una interacción entre los Casos de Uso y la arquitectura, los Casos de Uso deben encajar en la arquitectura cuando se llevan a cabo y la arquitectura debe permitir el desarrollo de todos los Casos de Uso requeridos, actualmente y en el futuro. Esto provoca que tanto arquitectura como Casos de Uso deban evolucionar en paralelo durante todo el proceso de desarrollo de software.

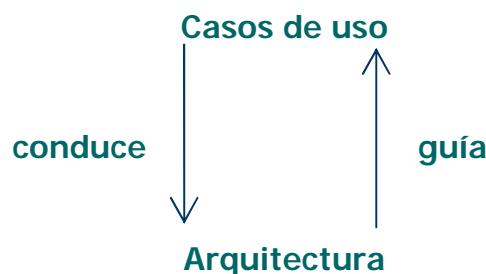


Figura 5: Interacción entre Casos de Uso y Arquitectura

En la Figura 6 se ilustra la evolución de la arquitectura durante las fases del Proceso Unificado. Se tiene una arquitectura más robusta en las fases finales del proyecto. En las fases iniciales lo que se hace es ir consolidando la arquitectura por medio de *líneas base* y se va modificando dependiendo de las necesidades del proyecto.

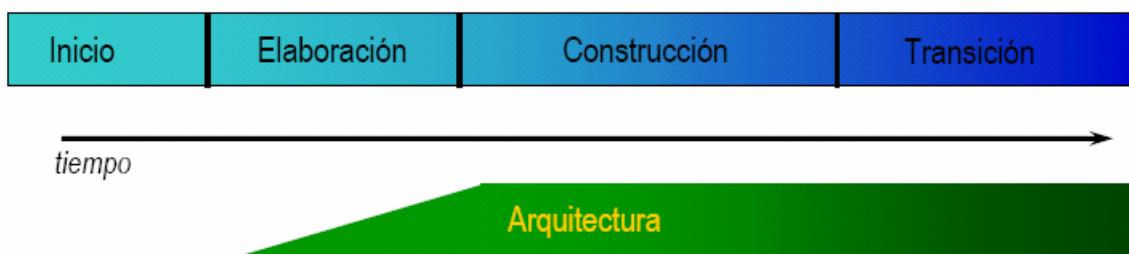


Figura 6: Evolución de la arquitectura del sistema.

Es conveniente ver el sistema desde diferentes perspectivas para comprender mejor el diseño por lo que la arquitectura se representa mediante varias vistas que se centran en aspectos concretos del sistema, abstrandose de los demás. Para el Proceso Unificado, todas las vistas juntas forman el llamado modelo 4+1 de la arquitectura [\[Kru95\]](#), el cual recibe este nombre porque lo forman las vistas lógica, de implementación, de proceso y de despliegue, más la de Casos de Uso que es la que da cohesión a todas.

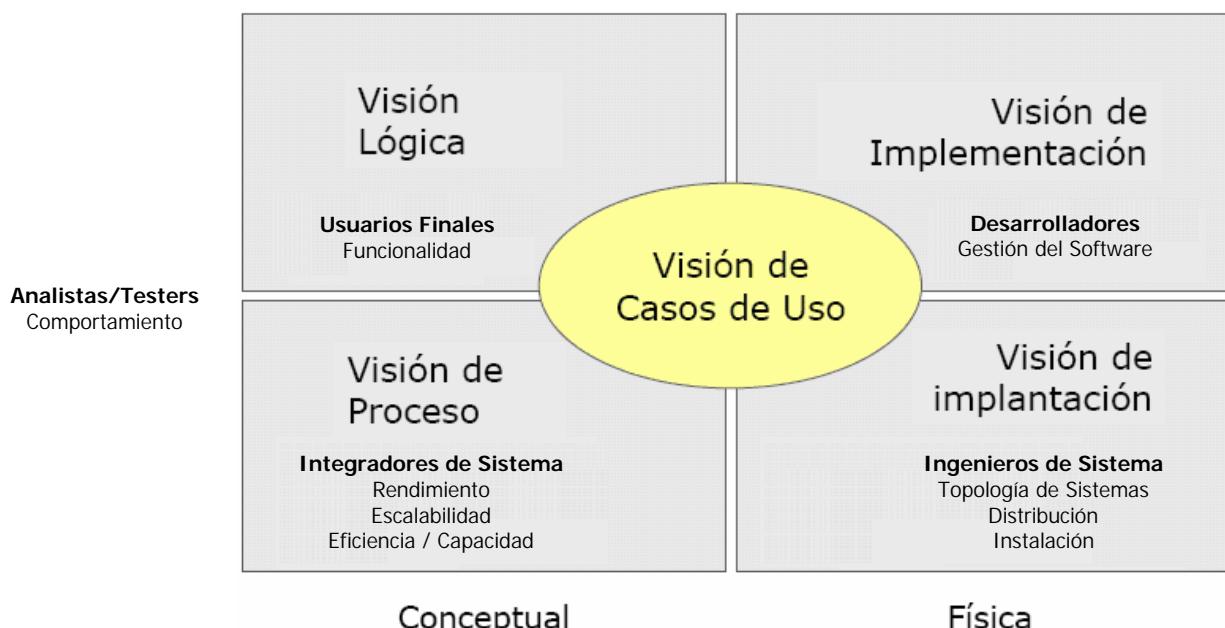


Figura 7: Modelo 4+1 de la arquitectura.

Al final de la **fase de elaboración** se obtiene una *línea base* de la arquitectura donde fueron seleccionados una serie de Casos de Uso arquitectónicamente relevantes (aquellos que ayudan a mitigar los riesgos más importantes, aquellos que son los más importantes para el usuario y aquellos que cubren las funcionalidades significativas).

Esta línea base es un esqueleto que:

- Representa una visión común a los desarrolladores
- Representa los actores y casos de uso más importantes
- Incluye los subsistemas e interfaces más importantes
- Modela las clases más relevantes
- Establece la arquitectura física del sistema
- Establece los objetos que se asignarán a cada nodo de procesamiento.

Durante la construcción los diversos modelos van desarrollándose hasta completarse. La descripción de la arquitectura sin embargo, no debería cambiar significativamente debido a que la mayor parte de la arquitectura se decidió durante la elaboración. Se incorporan pocos cambios a la arquitectura en las fases siguientes. [JBR00].

3.1.3. Proceso iterativo e incremental

El equilibrio correcto entre los Casos de Uso y la arquitectura es algo muy parecido al equilibrio de la forma y la función en el desarrollo del producto, lo cual se consigue con el tiempo. Para esto, la estrategia que se propone en el Proceso Unificado es tener un **proceso iterativo e incremental** en donde el trabajo se divide en partes más pequeñas o mini proyectos, permitiendo que el equilibrio entre Casos de Uso y arquitectura se vaya logrando durante cada mini proyecto, y se continúe evolucionando durante todo el proceso de desarrollo. Cada mini proyecto se puede ver como una iteración (un recorrido más o menos completo a lo largo de todos los flujos de trabajo fundamentales) del cual se obtiene un incremento que produce un crecimiento en el producto.

Una iteración puede realizarse por medio de una cascada como se muestra en la Figura 8. Se pasa por los flujos fundamentales (Requisitos, Análisis, Diseño, Implementación y Pruebas), también existe una **planificación** de la iteración, un **análisis** de la iteración y algunas actividades específicas de la iteración. Al finalizar se realiza una integración de los resultados con lo obtenido de las iteraciones anteriores.

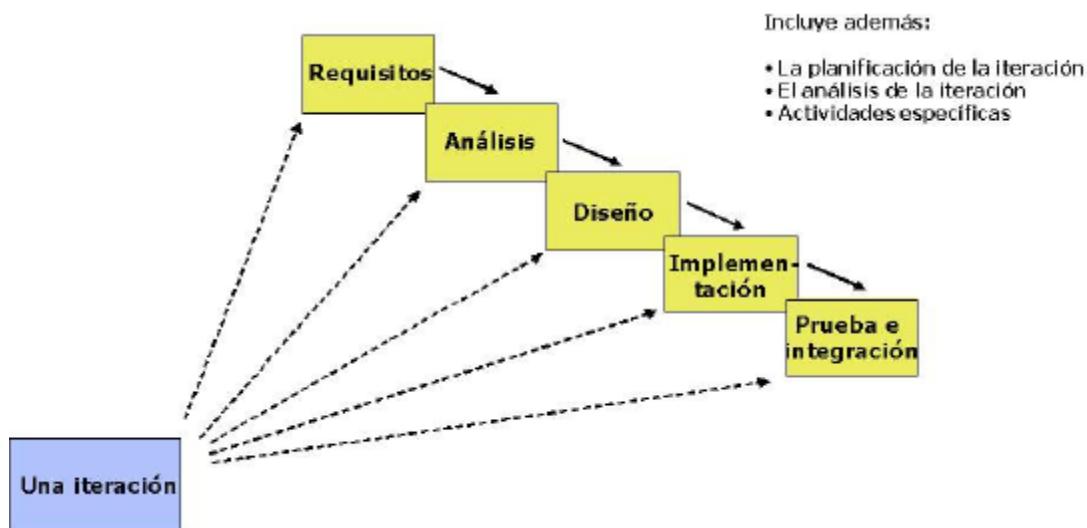


Figura 8: Una iteración del Proceso Unificado.

El proceso iterativo e incremental consta de una secuencia de iteraciones. Cada iteración aborda una parte de la funcionalidad total, pasando por todos los flujos de trabajo relevantes y refinando la arquitectura. Cada iteración se analiza cuando termina. Se puede determinar si han aparecido nuevos requisitos o han cambiado los existentes, afectando a las iteraciones siguientes. Durante la planificación de los detalles de la siguiente iteración, el equipo también examina cómo afectarán los riesgos que aún quedan al trabajo en curso. Toda la retroalimentación de la iteración pasada permite reajustar los objetivos para las siguientes iteraciones. Se continúa con esta dinámica hasta que se haya finalizado por completo con la versión actual del producto.

El Proceso Unificado divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en los distintas actividades. En la Figura 9 se muestra cómo varía el esfuerzo asociado a las disciplinas según la fase en la que se encuentre el proyecto de Proceso Unificado.

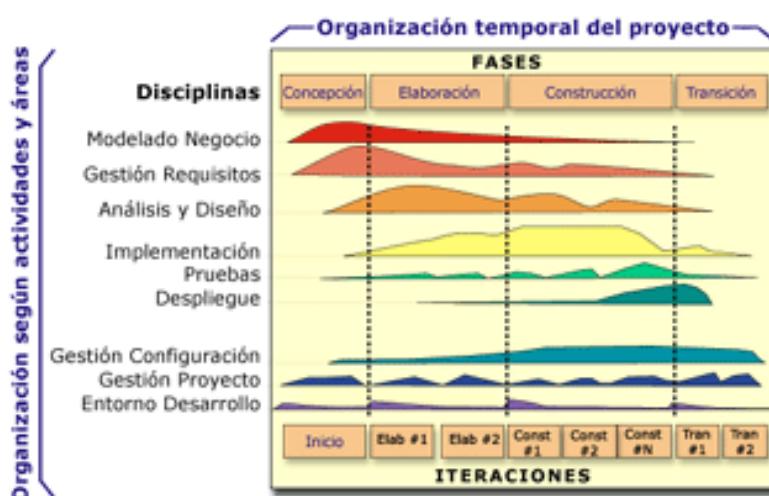


Figura 9: Esfuerzo en actividades según fase del proyecto, según Ciclo de Vida RUP v2003.

Las primeras iteraciones (en las fases de Inicio y Elaboración) se enfocan hacia la comprensión del problema y la tecnología, la delimitación del ámbito del proyecto, la eliminación de los riesgos críticos, y al establecimiento de una *línea base* de la arquitectura.

Durante la **fase de inicio** las iteraciones hacen poner mayor énfasis en actividades modelado del negocio y de requisitos.

En la **fase de elaboración**, las iteraciones se orientan al desarrollo de la *línea base* de la arquitectura, abarcan más los flujos de trabajo de requerimientos, modelo de negocios (refinamiento), análisis, diseño y una parte de implementación orientado a la *línea base* de la arquitectura.

En la **fase de construcción**, se lleva a cabo la construcción del producto por medio de una serie de iteraciones.

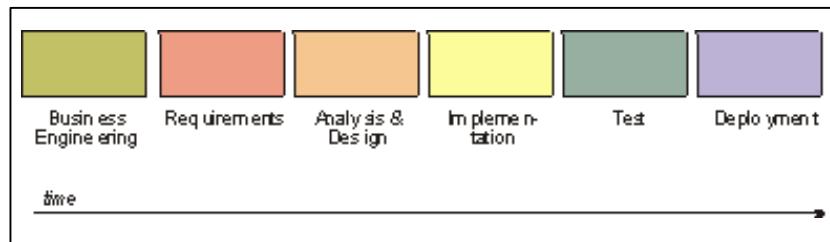
Para cada iteración se selecciona algunos Casos de Uso, se refina su análisis y diseño y se procede a su implementación y pruebas. Se realiza una pequeña cascada para cada ciclo. Se realizan tantas iteraciones hasta que se termine la implementación de la nueva versión del producto.

En la **fase de transición** se pretende garantizar que se tiene un producto preparado para su entrega a la comunidad de usuarios.

Como se puede observar en cada fase participan todas las disciplinas, pero que dependiendo de la fase el esfuerzo dedicado a una disciplina varía.

En la figura 10 se puede observar gráficamente la diferencia con el enfoque secuencial y el iterativo incremental y el grado de esfuerzo aplicado en cada especialidad del proceso varía de acuerdo con las iteraciones y fases.

Enfoque Secuencial



Enfoque Iterativo e Incremental

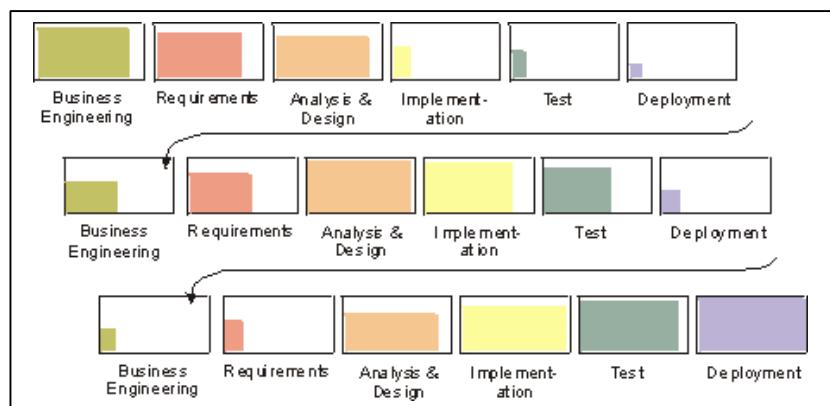


Figura 10: Diferencia entre el clásico enfoque secuencial y el enfoque Iterativo e Incremental.

3.2. Principios Esenciales

- Atacar los principales riesgos de forma temprana y realizar una revisión continua...o éstos atacarán el proyecto.
- Asegurarse de entregar valor al cliente.
- Enfocarse en software ejecutable.

- Incluir al *cambio* en las fases tempranas del proyecto.
- Establecer una línea base de arquitectura en las fases tempranas.
- Construir el sistema con componentes.
- Trabajar de forma conjunta, como un equipo.
- Hacer que la calidad sea una forma de encarar el proyecto, no algo posterior.

3.3. Visión General de Fases, Objetivos e Hitos.

El Proceso Unificado se repite a lo largo de una serie de ciclos que constituyen la vida de un producto. Cada ciclo concluye con una generación del producto para los clientes. Cada ciclo consta de cuatro fases: Inicio, Elaboración, Construcción y Transición. Cada fase se subdivide a la vez en iteraciones, el número de iteraciones en cada fase es variable.

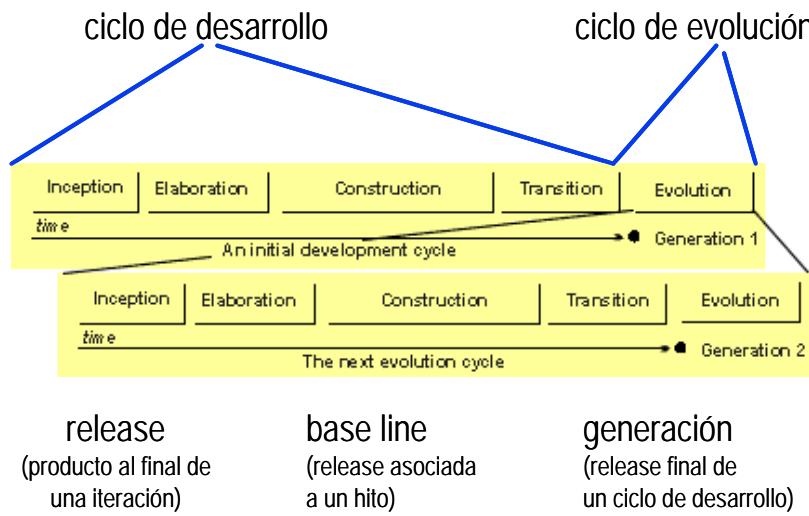


Figura 11: Ciclos, releases y *linea base*.

Cada fase se concluye con un hito bien definido, un punto en el tiempo en el cual se deben tomar ciertas decisiones críticas y alcanzar las metas clave antes de pasar a la siguiente fase, ese hito principal de cada fase se compone de hitos menores que podrían ser los criterios aplicables a cada iteración.

Los hitos para cada una de las fases son:

- Inicio – Objetivos del Ciclo de Vida
- Elaboración – Arquitectura del Ciclo de Vida
- Construcción – Capacidad Operativa Inicial
- Transición – Release del producto

Las fases y sus respectivos hitos se ilustran en la Figura 12.

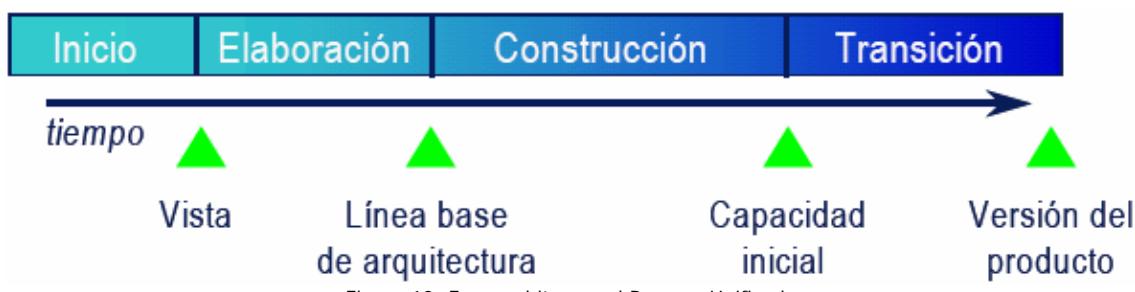
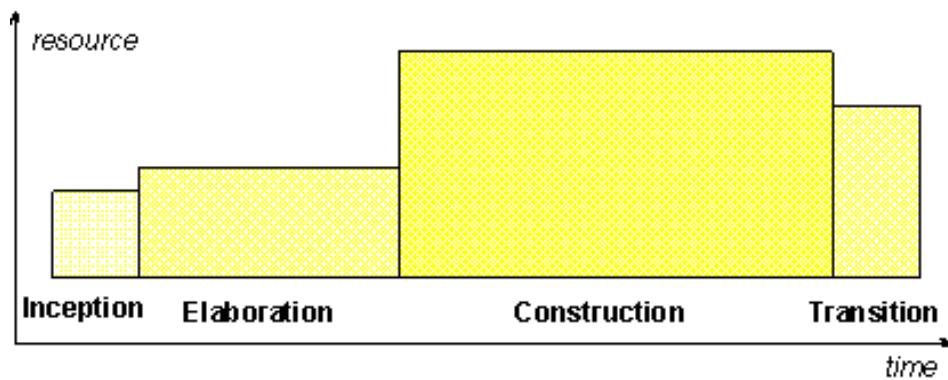


Figura 12: Fases e hitos en el Proceso Unificado

La duración y esfuerzo dedicado en cada fase es variable dependiendo de las características del proyecto. Sin embargo, la Figura 13 ilustra porcentajes frecuentes al respecto. Consecuentemente con el esfuerzo señalado, la Figura 14 ilustra una distribución típica de recursos humanos necesarios a lo largo del proyecto.

	Inicio	Elaboración	Construcción	Transición
Esfuerzo	5 %	20 %	65 %	10%
Tiempo Dedicado	10 %	30 %	50 %	10%

Figura 13: Distribución típicas de esfuerzo y tiempo



3.4. Mapa Conceptual del Proceso Unificado

En la figura 15 se presenta un mapa conceptual que resume lo expuesto anteriormente sobre las etapas y fases del Proceso Unificado.

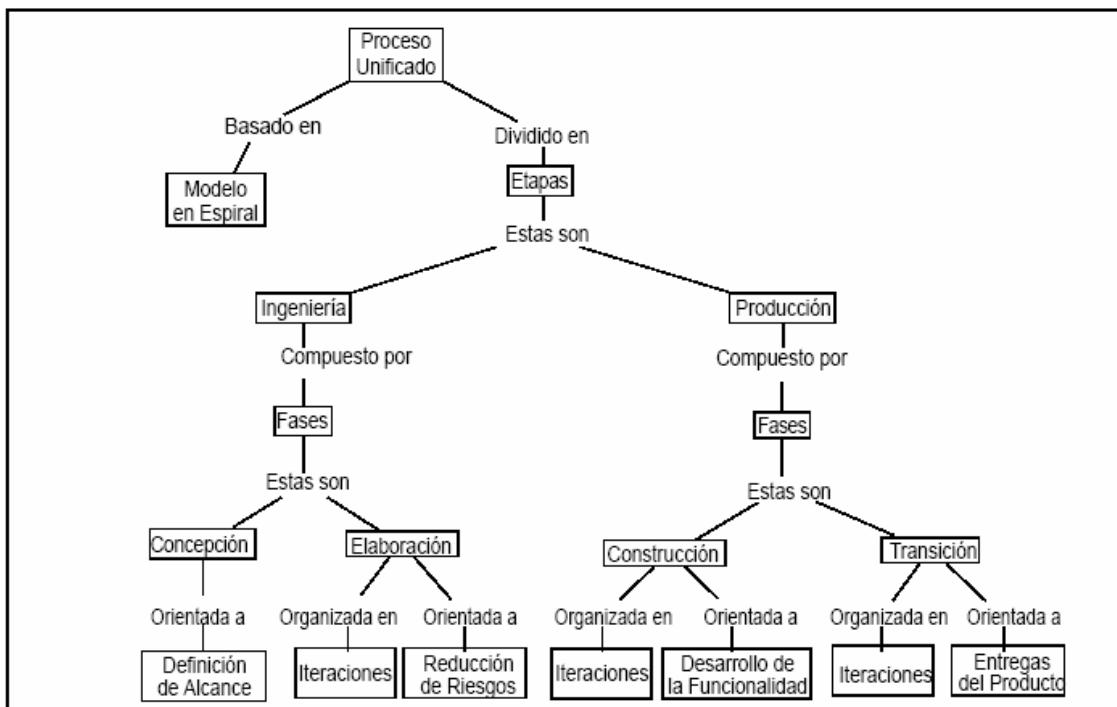


Figura 15: Mapa conceptual de etapas y fases del Proceso Unificado.

3.5. Detalle de Fases, Objetivos e Hitos

3.5.1. Fase de Inicio

Durante la fase de inicio se define el modelo del negocio y el alcance del proyecto. Se identifican todos los actores y Casos de Uso, y se diseñan los Casos de Uso más esenciales (aproximadamente el 20% del modelo completo). Se desarrolla, un plan de negocio para determinar que recursos deben ser asignados al proyecto.

3.5.1.1. Objetivo

- Establecer el ámbito del proyecto y sus límites.
- Encontrar los Casos de Uso críticos del sistema, los escenarios básicos que definen la funcionalidad.
- Mostrar al menos una arquitectura candidata para los escenarios principales.
- Estimar el coste en recursos y tiempo de todo el proyecto.
- Estimar los riesgos, las fuentes de incertidumbre.

3.5.1.2. Hito

- Objetivo del Ciclo de Vida (LCO, Lifecycle Objectives)
 - Un documento de [Visión](#): Una visión general de los requerimientos del proyecto, características clave y restricciones principales.
 - [Plan de Desarrollo del Software](#), mostrando fases e iteraciones.
 - [Modelo de Negocio](#), si es necesario.
 - El [Caso Uso del Negocio](#).
 - Modelo inicial de [Casos de Uso](#) (10-20% completado).
 - Un [Glosario](#) inicial: Terminología clave del dominio.
 - [Lista de Riesgos](#) y Plan de Contingencia.
 - Prototipos exploratorios para probar conceptos o la arquitectura candidata.

3.5.1.3. Criterio de Evaluación

- Todos los interesados en el proyecto coinciden en la definición del ámbito del sistema y las estimaciones de agenda.
- Entendimiento de los requisitos, como evidencia de la fidelidad de los Casos de Uso principales.
- Las estimaciones de tiempo, coste y riesgo son creíbles.
- Comprensión total de cualquier prototipo de la arquitectura desarrollado.
- Los gastos hasta el momento se asemejan a los planeados.

Si el proyecto no pasa estos criterios hay que plantearse abandonarlo o repensarlo profundamente.

3.5.2. Fase de Elaboración

El propósito de la fase de elaboración es analizar el dominio del problema, establecer los cimientos de la arquitectura, desarrollar el plan del proyecto y eliminar los mayores riesgos.

En esta fase se construye un prototipo de la arquitectura, que debe evolucionar en iteraciones sucesivas hasta convertirse en el sistema final. Este prototipo debe contener los Casos de Uso críticos identificados en la fase de inicio. También debe demostrarse que se han evitado los riesgos más graves.

En esta fase se debe tratar de abarcar todo el proyecto con la profundidad mínima. Sólo se profundiza en los puntos críticos de la arquitectura o riesgos importantes.

En esta fase se actualizan todos los productos de la fase de inicio.

3.5.2.1. Objetivo

- Definir, validar y cimentar la arquitectura.
- Completar la Visión.
- Crear un plan fiable para la fase de construcción. Este plan puede evolucionar en sucesivas iteraciones. Debe incluir los costes si procede.
- Demostrar que la arquitectura propuesta soportará la visión con un coste razonable y en un tiempo razonable.

3.5.2.2. Hito

- Arquitectura del Ciclo de Vida (LCA, Lifecycle Architecture)
 - Un modelo de [Casos de Uso](#) completo al menos hasta el 80%: todos los casos y actores identificados, la mayoría de los casos desarrollados.
 - Requisitos adicionales que capturan los **requisitos no funcionales** y cualquier requisito no asociado con un Caso de Uso específico.
 - Descripción de la arquitectura software.
 - Un prototipo ejecutable de la arquitectura.
 - Documentos de [Visión](#), [Lista de Riesgos](#) y [Caso de Negocio](#) revisados.
 - Plan de Desarrollo para el proyecto.
 - Un caso de desarrollo actualizado que especifica el proceso a seguir.
 - Un Manual de Usuario preliminar (opcional).

3.5.2.3. Criterio de Evaluación

- La visión del producto es estable.
- La arquitectura es estable.
- Se ha demostrado mediante la ejecución del prototipo que los principales elementos de riesgo han sido abordados y resueltos.
- El plan para la fase de construcción es detallado y preciso. Las estimaciones son creíbles.
- Todos los interesados coinciden en que la visión actual será alcanzada si se siguen los planes actuales en el contexto de la arquitectura actual.
- Los gastos hasta ahora son aceptables, comparados con los previstos.

Si no se superan los criterios de evaluación quizá sea necesario abandonar el proyecto o replanteárselo considerablemente.

3.5.3. Fase de Construcción

La finalidad principal de esta fase es alcanzar la capacidad operacional del producto de forma incremental a través de las sucesivas iteraciones. Durante esta fase todos los componentes, características y requisitos deben ser implementados, integrados y probados en su totalidad, obteniendo una versión aceptable del producto.

3.5.3.1. Objetivo

- Minimizar los costes de desarrollo mediante la optimización de recursos y evitando el tener que rehacer un trabajo o incluso desecharlo.
- Conseguir una calidad adecuada tan rápido como sea práctico.
- Conseguir versiones funcionales (alfa, beta, y otras versiones de prueba) tan rápido como sea práctico.

3.5.3.2. Hito

- Capacidad Operativa Inicial (IOC, Initial Operational Capability)
 - Modelos Completos ([Caso de Uso](#), Análisis, Diseño, Despliegue e Implementación).
 - Arquitectura íntegra (mantenida y mínimamente actualizada).
 - Riesgos Presentados Mitigados.
 - Plan del Proyecto para la fase de Transición.
 - Manual Inicial de Usuario (con suficiente detalle).
 - Prototipo Operacional – beta.
 - [Caso de Uso del Negocio](#) actualizado.

3.5.3.3. Criterio de Evaluación

- El producto es estable y maduro como para ser entregado a la comunidad de usuario para ser probado.
- Todos los usuarios expertos están listos para la transición en la comunidad de usuarios.
- Son aceptables los gastos actuales versus los gastos planeados.
- La arquitectura es estable.

3.5.4. Fase de Transición

La finalidad de la fase de transición es poner el producto en manos de los usuarios finales, para lo que se requiere desarrollar nuevas versiones actualizadas del producto, completar la documentación, entrenar al usuario en el manejo del producto, y en general tareas relacionadas con el ajuste, configuración, instalación y facilidad de uso del producto.

Estos son algunos de los elementos que se pueden incluir esta fase:

- Prueba de la versión Beta para validar el nuevo sistema frente a las expectativas de los usuarios
- Funcionamiento paralelo con los sistemas legados que están siendo sustituidos por nuestro proyecto.
- Conversión de las bases de datos operacionales.
- Entrenamiento de los usuarios y técnicos de mantenimiento.
- Traspaso del producto a los equipos de marketing, distribución y venta.

3.5.4.1. Objetivo

- Conseguir que el usuario se valga por si mismo.
- Un producto final que cumpla los requisitos esperados, que funcione y satisfaga suficientemente al usuario.

3.5.4.2. Hito

- Release del Producto (PR, Product Release)
 - Prototipo Operacional
 - Documentos Legales
 - Caso del Negocio Completo
 - Línea de Base del Producto completa y corregida que incluye todos los modelos del sistema
 - Descripción de la Arquitectura completa y corregida
 - Las iteraciones de esta fase irán dirigidas normalmente a conseguir una nueva versión.

3.5.4.3. Criterio de Evaluación

- El usuario se encuentra satisfecho.
- Son aceptables los gastos actuales versus los gastos planificados.

3.5.5. Grado de Finalización de los Artefactos según la Fase

En la figura 16 se aprecia la evolución que sufren los artefactos, agrupados según la disciplina a la que pertenecen, a través de las diferentes fases del proceso.

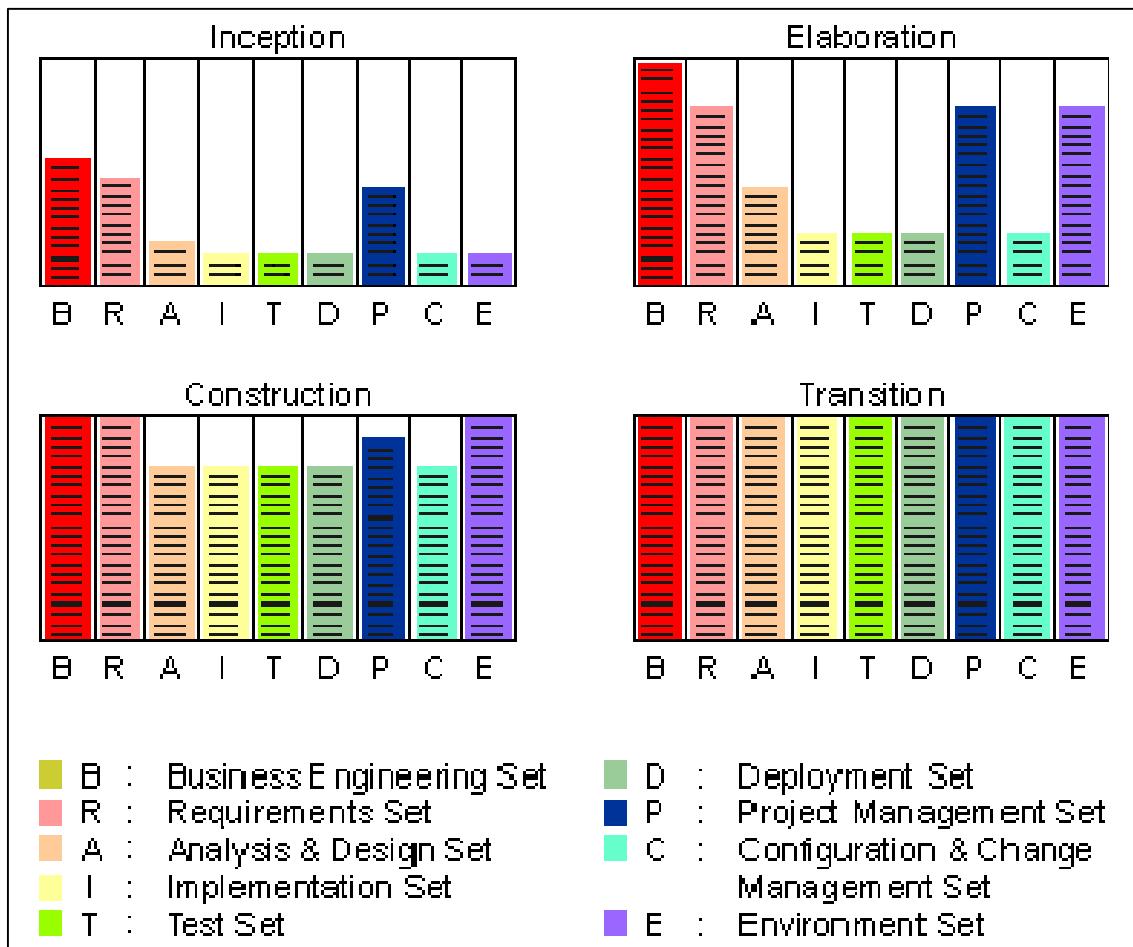


Figura 16: Grado de finalización de los artefactos a medida que se avanza en las fases.

3.6. Roles. El Quién.

Un rol define el comportamiento y responsabilidades de un individuo, o de un grupo de individuos trabajando juntos como un equipo. Una persona puede desempeñar diversos roles, así como un mismo rol puede ser representado por varias personas.

Las responsabilidades de un rol son tanto el llevar a cabo un conjunto de actividades como el ser el dueño de un conjunto de artefactos.

EL Proceso Unificado define grupos de roles, agrupados por participación en actividades relacionadas. Estos grupos son:

- Analistas
 - Analista de procesos de negocio.
 - Diseñador del negocio.
 - Analista de sistema.
 - Especificador de requisitos.
- Desarrolladores
 - Arquitecto de software.
 - Diseñador
 - Diseñador de interfaz de usuario
 - Diseñador de cápsulas.
 - Diseñador de base de datos.
 - Implementador.
 - Integrador.
- Gestores
 - Jefe de proyecto
 - Jefe de control de cambios.
 - Jefe de configuración.
 - Jefe de pruebas
 - Jefe de despliegue
 - Ingeniero de procesos
 - Revisor de gestión del proyecto
 - Gestor de pruebas.
- Apoyo
 - Documentador técnico
 - Administrador de sistema
 - Especialista en herramientas
 - Desarrollador de cursos
 - Artista gráfico
- Especialista en pruebas
 - Especialista en Pruebas (tester)
 - Analista de pruebas
 - Diseñador de pruebas
- Otros roles
 - Stakeholders.
 - Revisor.
 - Coordinación de revisiones

PROYECTO FINAL TC1/TD

- Revisor técnico
- Cualquier rol

3.7. Actividades. El Cómo.

Una actividad en concreto es una unidad de trabajo que una persona que desempeñe un rol puede ser solicitado a que realice. Las actividades tienen un objetivo concreto, normalmente expresado en términos de crear o actualizar algún producto.

Las actividades pueden completarse siguiendo pasos preestablecidos, que pueden agruparse en:

- Pasos de **Concepción**, en los que la persona interpretando el rol comprende la naturaleza de la tarea, recolecta y examina los artefactos de entrada y formula el resultado esperado.
- Pasos de **Ejecución**, en los que el rol crea o actualiza algunos artefactos.
- Pasos de **Revisión**, en los que el rol inspecciona los resultados contra ciertos criterios.

3.8. Artefactos. El Qué.

Un producto o artefacto es un trozo de información que es producido, modificado o usado durante el proceso de desarrollo de software. Los productos son los resultados tangibles del proyecto, las cosas que va creando y usando hasta obtener el producto final.

Un artefacto puede ser cualquiera de los siguientes:

- Un documento, como el documento de la arquitectura del software.
- Un modelo, como el modelo de Casos de Uso o el modelo de diseño.
- Un elemento del modelo, un elemento que pertenece a un modelo como una clase, un Caso de Uso o un subsistema.

En la sección [Artefactos entregables](#), se explican brevemente en qué consiste cada uno de los artefactos requeridos para la carpeta de proyecto.

3.9. Disciplinas. El Cuándo.

3.9.1. Modelado del Negocio

Con este flujo de trabajo pretendemos llegar a un mejor entendimiento de la organización donde se va a implantar el producto.

3.9.1.1. Objetivo

- Entender la estructura y la dinámica de la organización para la cual el sistema va ser desarrollado (organización objetivo).
- Entender el problema actual en la organización objetivo e identificar potenciales mejoras.
- Asegurar que clientes, usuarios finales y desarrolladores tengan un entendimiento común de la organización objetivo.
- Derivar los requisitos del sistema necesarios para apoyar a la organización objetivo.

Para lograr estos objetivos, el modelo de negocio describe como desarrollar una visión de la nueva organización, basado en esta visión se definen procesos, roles y responsabilidades de la organización por medio de un modelo de Casos de Uso del negocio y un Modelo de Objetos del Negocio. Complementario a estos modelos, se desarrollan otras especificaciones tales como un Glosario.

3.9.2. Gestión de Requerimientos

Este es uno de los flujos de trabajo más importantes, porque en él se establece qué tiene que hacer exactamente el sistema que construyamos. En esta línea los requisitos son el contrato que se debe cumplir, de modo que los usuarios finales tienen que comprender y aceptar los requisitos que especifiquemos.

3.9.2.1. Objetivo

- Establecer y mantener un acuerdo entre clientes y otros stakeholders sobre lo que el sistema podría hacer.
- Proveer a los desarrolladores un mejor entendimiento de los requisitos del sistema.
- Definir el ámbito del sistema.
- Proveer una base para la planeación de los contenidos técnicos de las iteraciones.
- Proveer una base para estimar costos y tiempo de desarrollo del sistema.
- Definir una interfaz de usuarios para el sistema, enfocada a las necesidades y metas del usuario.

Los requisitos se dividen en dos grupos. Los requisitos funcionales representan la funcionalidad del sistema. Se modelan mediante diagramas de Casos de Uso. Los requisitos no funcionales representan aquellos atributos que debe exhibir el sistema, pero que no son una funcionalidad específica. Por ejemplo requisitos de facilidad de uso, fiabilidad, eficiencia, portabilidad, etc.

Para capturar los requisitos es preciso entrevistar a todos los interesados en el proyecto, no sólo a los usuarios finales, y anotar todas sus peticiones. A partir de ellas hay que descubrir lo que necesitan y expresarlo en forma de requisitos.

En este flujo de trabajo, y como parte de los requisitos de facilidad de uso, se diseña la interfaz gráfica de usuario. Para ello habitualmente se construyen prototipos de la interfaz gráfica de usuario que se contrastan con el usuario final.

3.9.3. Análisis & Diseño

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

3.9.3.1. Objetivo

- Transformar los requisitos al diseño del futuro sistema.
- Desarrollar una arquitectura para el sistema.
- Adaptar el diseño para que sea consistente con el entorno de implementación, diseñando para el rendimiento.

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.

Al principio de la fase de elaboración 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 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 de software, que captura varias vistas arquitectónicas del sistema.

3.9.4. Implementación

En este flujo de trabajo se implementan las clases y objetos en archivos fuente, binarios, ejecutables y demás. Además se deben hacer las pruebas de unidad: cada implementador es responsable de probar las unidades que produzca. El resultado final de este flujo de trabajo es un sistema ejecutable.

3.9.4.1. Actividades

- Planificar qué subsistemas deben ser implementados y en qué orden deben ser integrados, formando el Plan de Integración.
- Cada implementador decide en qué orden implementa los elementos del subsistema.
- Si encuentra errores de diseño, los notifica.
- Se prueban los subsistemas individualmente.
- Se integra el sistema siguiendo el plan.

La estructura de todos los elementos implementados forma el modelo de implementación. La integración debe ser incremental, es decir, en cada momento sólo se añade un elemento. De este modo es más fácil localizar fallos y los componentes se prueban más a fondo. En fases tempranas del proceso se pueden implementar prototipos para reducir el riesgo. Su utilidad puede ir desde ver si el sistema es viable desde el principio, probar tecnologías o diseñar la interfaz de usuario. Los prototipos pueden ser exploratorios (desechables) o evolutivos. Estos últimos llegan a transformarse en el sistema final.

3.9.5. Pruebas

Este flujo de trabajo es el encargado de evaluar la calidad del producto que estamos desarrollando, pero no para aceptar o rechazar el producto al final del proceso de desarrollo, sino que debe ir integrado en todo el ciclo de vida.

3.9.5.1. Objetivo

- Encontrar y documentar defectos en la calidad del software.
- Generalmente asesora sobre la calidad del software percibida.
- Provee la validación de los supuestos realizados en el diseño y especificación de requisitos por medio de demostraciones concretas.
- Verificar las funciones del producto de software según lo diseñado.
- Verificar que los requisitos tengan su apropiada implementación.

Las actividades de este flujo comienzan pronto en el proyecto con el plan de prueba (el cual contiene información sobre los objetivos generales y específicos de la prueba en el proyecto, así como las estrategias y recursos con que se dotará a esta tarea), o incluso antes con alguna evaluación durante la fase de inicio, y continuará durante todo el proyecto.

El desarrollo del flujo de trabajo consistirá en planificar qué es lo que hay que probar, diseñar cómo se va a hacer, implementar lo necesario para llevarlos a cabo, ejecutarlos en los niveles necesarios y obtener los resultados, de forma que la información obtenida nos sirva para ir refinando el producto a desarrollar.

3.9.6. Despliegue

El objetivo de este flujo de trabajo es producir con éxito distribuciones del producto y distribuirlo a los usuarios

3.9.6.1. Actividades

- Probar el producto en su entorno de ejecución final.
- Empaquetar el software para su distribución.
- Distribuir el software.
- Instalar el software.
- Proveer asistencia y ayuda a los usuarios.
- Formar a los usuarios y al cuerpo de ventas.
- Migrar el software existente o convertir bases de datos.

Este flujo de trabajo se desarrolla con mayor intensidad en la fase de transición, ya que el propósito del flujo es asegurar una aceptación y adaptación sin complicaciones del software por parte de los usuarios. Su ejecución inicia en fases anteriores, para preparar el camino, sobre todo con actividades de planificación, en la elaboración del manual de usuario y tutoriales.

3.9.7. Gestión de Proyecto

La Gestión del proyecto es el arte de lograr un balance al gestionar objetivos, riesgos y restricciones para desarrollar un producto que sea acorde a los requisitos de los clientes y los usuarios.

3.9.7.1. Objetivo

- Proveer un marco de trabajo para la gestión de proyectos de software intensivos.
- Proveer guías prácticas realizar planeación, contratar personal, ejecutar y monitorear el proyecto.
- Proveer un marco de trabajo para gestionar riesgos.

La planeación de un proyecto posee dos niveles de abstracción: un plan para las fases y un plan para cada iteración.

3.9.8. Gestión de Cambios (Change management)

La finalidad de este flujo de trabajo es mantener la integridad de todos los artefactos que se crean en el proceso, así como de mantener información del proceso evolutivo que han seguido.

3.9.9. **Entorno**

La finalidad de este flujo de trabajo es dar soporte al proyecto con las adecuadas herramientas, procesos y métodos. Brinda una especificación de las herramientas que se van a necesitar en cada momento, así como definir la instancia concreta del proceso que se va a seguir.

3.9.9.1. Objetivo

- Selección y adquisición de herramientas
- Establecer y configurar las herramientas para que se ajusten a la organización.
- Configuración del proceso.
- Mejora del proceso.
- Servicios técnicos.

El principal artefacto que se usa en este flujo de trabajo es el caso de desarrollo que especifica para el proyecto actual en concreto, como se aplicará el proceso, que productos se van a utilizar y como van a ser utilizados. Además se tendrán que definir las guías para los distintos aspectos del proceso, como pueden ser el modelado del negocio y los Casos de Uso, para la interfaz de usuario, el diseño, la programación, el manual de usuario.

4. Artefactos entregables

A continuación se indican y describen cada uno de los artefactos que serán generados y utilizados por el proyecto y que constituyen los entregables. Esta lista constituye la configuración del Proceso Unificado desde la perspectiva de artefactos, y que proponemos para este proyecto.

Es preciso destacar que de acuerdo a la filosofía del Proceso Unificado (y de todo proceso iterativo e incremental), todos los artefactos son objeto de modificaciones a lo largo del proceso de desarrollo, con lo cual, **sólo al término del proceso podríamos tener una versión definitiva y completa de cada uno de ellos**, como se explicó en el [apartado correspondiente](#). Sin embargo, el resultado de cada iteración y los hitos del proyecto están enfocados a conseguir un cierto grado de completitud y estabilidad de los artefactos. Esto fue indicado anteriormente cuando se presentaron los [objetivos de cada iteración](#).

4.1. Descripción de la Empresa

Cátedra	Carácter
ASA	Requerido

Esta sección pretende servir de introducción al proyecto de desarrollo a realizar. Conociendo las características generales de la empresa, las personas involucradas en el proyecto pueden conocer mejor el escenario en el que se desenvolverán.

Se pueden incluir aquí la historia de la empresa, cuando fue fundada, evolución, constitución legal, quiénes son los accionistas de la empresa y cuál ha sido su evolución; sus instalaciones y localidades donde tiene operaciones, entre otros.

4.2. Plan de Desarrollo del Software

Cátedra	Carácter
ASA	Opcional

Este documento contiene la definición del plan de desarrollo a seguir, estableciendo una visión general del proyecto, la organización y planificación temporal, descripción de las fases e iteraciones y gestión del proceso.

Se puede visualizar un ejemplo de este artefacto en la sección [Plan Desarrollo del Software](#) del proyecto modelo.

4.3. Visión

Cátedra	Carácter
ASA	Opcional

Este documento define la visión del producto desde la perspectiva del cliente, especificando las necesidades y características del producto. Constituye una base de acuerdo en cuanto a los requisitos del sistema.

Se puede visualizar un ejemplo de este artefacto en la sección [Documento Visión](#) del proyecto modelo.

4.4. Modelo de Casos de Uso del Negocio

Cátedra	Carácter
ASA	Opcional

Es un modelo de las funciones de negocio vistas desde la perspectiva de los actores externos (Agentes de registro, solicitantes finales, otros sistemas etc.). Permite situar al sistema en el contexto

PROYECTO FINAL TC1/TD

organizacional haciendo énfasis en los objetivos en este ámbito. Este modelo se representa con un Diagrama de Casos de Uso usando estereotipos específicos para este modelo.

Se puede visualizar un ejemplo de este artefacto en la sección [Casos de Uso del Negocio](#) del proyecto modelo.

4.5. Modelo de Dominio / Objetos del Negocio

Cátedra	Carácter
ASA	Opcional

Es un modelo que describe la realización de cada caso de uso del negocio, estableciendo los actores internos, la información que en términos generales manipulan y los flujos de trabajo (workflows) asociados al caso de uso del negocio. Para la representación de este modelo se utilizan Diagramas de Colaboración (para mostrar actores externos, internos y las entidades (información) que manipulan, un Diagrama de Clases para mostrar gráficamente las entidades del sistema y sus relaciones, y Diagramas de Actividad para mostrar los flujos de trabajo.

Se puede visualizar un ejemplo de este artefacto en la sección [Modelo de Objetos del Negocio](#) del proyecto modelo.

4.6. Estudio de Viabilidad

Toda decisión de desarrollar un software debe responder a un estudio previo de ventajas y desventajas asociadas a su construcción e implementación, la profundidad con que se realice dependerá de lo que aconseje cada proyecto en particular.

En términos generales, seis son los estudios particulares que deben realizarse para evaluar un proyecto de software: los de viabilidad legal, técnica, operativa, de gestión, comercial y de impacto económico/financiero. Cualquiera de ellos que llegue a una conclusión negativa determinará que el proyecto no se lleve a cabo, aunque razones estratégicas, comerciales, humanitarias u otras de índole subjetivas podrían hacer recomendable una opción que no sea viable financiera o económica.

Por lo regular, el estudio de una inversión en el desarrollo de software se centra en la viabilidad financiera o económica, y toma al resto de las variables únicamente como referencia.

4.6.1. Viabilidad Legal

Cátedra	Carácter
TC1	Requerido

Esta viabilidad se refiere a la necesidad de determinar la inexistencia de trabas legales para la instalación y operación normal del software.

Un proyecto de software puede ser viable tanto por tener un mercado asegurado como por ser técnicamente factible. Sin embargo, podrían existir algunas restricciones de carácter legal que impedirían su funcionamiento en los términos que se pudiera haber previsto, no haciendo recomendable su ejecución.

Características del estudio de viabilidad legal que debe ser incluido en la carpeta:

- Descripción de los impedimentos legales que pueden alcanzar este proyecto.
- De no haberlos, declaración de no existencia de impedimentos legales.
- De haberlos, y disponer de copia de la legislación, colocarlos en la carpeta como un apéndice.

4.6.2. Viabilidad Técnica

PROYECTO FINAL TC1/TD

Cátedra	Carácter
TC1	Requerido

Estudia las posibilidades materiales y físicas de producir el software que desea generarse con el proyecto. Tiene por objeto proveer información para cuantificar el monto de las inversiones y de los costos de operación. En él se revisan aspectos relativos al hardware, software y telecomunicaciones y aquellos referidos a la cantidad y capacitación del personal.

En caso de ser insuficiente el parque tecnológico y requiera de su ampliación, la misma debe estar detallada en este punto y deberá ser considerada en el momento de realizar el estudio de viabilidad económica ya que pasa a formar parte de la inversión necesaria para que el sistema pueda funcionar adecuadamente. Dentro de este estudio se incluye el nivel de capacitación del personal que utilizará el sistema

Características del estudio de viabilidad técnica que debe ser incluido en la carpeta:

- Análisis del parque instalado (de acuerdo a los datos recogidos en el relevamiento detallado)
- Determinación de necesidad de actualización, mejora o ampliación del parque instalado con el detalle de que se debe incorporar.
- Grado de necesidad de capacitación de los usuarios.
- Requerimientos de instalaciones (redes, telecomunicaciones, adecuaciones de los ámbitos de trabajo, etc.) considerados necesarios para la implementación del sistema.

4.6.3. Viabilidad Operativa

Cátedra	Carácter
TC1	Requerido

Dependiendo de las características del proyecto y de la modalidad con que la empresa encaró el mismo, los aspectos organizacionales pueden haber sido evaluados al definirse la estrategia de sistemas, en cuyo caso, no se requerirá la ejecución de esta fase.

En caso contrario, se deberá efectuar el estudio correspondiente que tiene por objeto evaluar el impacto del proyecto sobre la organización.

Para ver con mayor claridad la importancia del estudio de Viabilidad operativo recurriremos a un ejemplo:

Tomemos el caso de un banco que en la actualidad maneja el conjunto de sus operaciones en forma tradicional cuyos sistemas tienen un gran porcentaje de procesamiento manual y en algunos casos batch (en la casa matriz), que decide utilizar tecnología de punta, y automatizar el grueso de sus operaciones, incorporar cajeros automáticos, conectar todos sus sistemas a una red de comunicaciones, y distribuir capacidad de procesamiento para sus 50 sucursales.

Desde el punto de vista técnico y económico, el proyecto resulta a priori, viable, dado que este tipo de instalación está ampliamente aprobada por otras entidades financieras, los beneficios que resultan de un proyecto de inversión de esta naturaleza son de fácil comprobación no hay más que consultar a las otras empresas y además las compañías proveedoras de equipos ya han adquirido experiencia suficiente, con un gran parque instalado y hasta existe la posibilidad de adquirir sistemas preplaneados de buen rendimiento y bajo costo.

En este caso el estudio de **Viabilidad Operativa** adquiere, su mayor significación, ya que la viabilidad del sistema no está en duda, pero lo que falta verificar es si este Banco, está en condiciones de absorberlo; ya que, el personal no tiene experiencia en el manejo de este tipo de sistemas, los gerentes verán modificados los procedimientos del conjunto de su operatoria y además deberán estar a la cabeza del cambio. Surgirán además los temores de una racionalización, además de la resistencia natural al cambio en todos los niveles; la operatoria de cada sucursal sufrirá profundas modificaciones, se requerirá la capacitación del personal en gran escala; la participación como usuario

PROYECTO FINAL TC1/TD

(una nueva función) en la definición de los futuros sistemas, podrán surgir modificaciones en la estructura de poder de la organización que tendrán relación directa con la reestructuración orgánica, etc. y además de todo esto se deberá seguir cumpliendo con las actividades diarias para que el Negocio siga funcionando.

Como vemos un cambio de esta envergadura puede generar una sucesión interminable de conflictos que si se dejan librados al azar las posibilidades de éxito del proyecto tendrán un alto grado de aleatoriedad.

Como vemos no se puede relativizar el estudio operativo y a efectos de reducir los márgenes de riesgo se deberá:

- Establecer el alcance de los cambios organizacionales
- Evaluar las normas, métodos y funciones organizacionales vigentes.
- Evaluar el desarrollo organizativo alcanzado.
- Analizar las relaciones de poder actuales y futuras y su efecto sobre el proyecto.
- Trazar una hipótesis de conflictos potenciales.
- Efectuar un análisis sobre la oportunidad del proyecto.

4.6.4. Viabilidad Económico/Financiera

Cátedra	Carácter
TC1	Requerido

Desde el punto de vista económico, aquí se analizan los recursos necesarios para llevar adelante el proyecto y se establece la cantidad de dinero necesario para llevar adelante el emprendimiento. La asignación de recursos a tareas y su utilización en el tiempo de vida del proyecto son parte de este estudio. Para representarlos utilizaremos los diagramas de GANTT y PERT.

Al momento de calcular los costos no debe olvidarse que, en un proyecto de este tipo, la mayor parte de los recursos son recursos humanos (analistas, programadores, diseñadores de bases de datos, etc.). Y que cada uno tiene un valor distinto de su hora/hombre dependiendo de su grado de especialización. **Es incorrecto que la hora de trabajo de un analista tenga el mismo valor que la de un programador o la de la persona que hizo el relevamiento.**

Características del estudio de viabilidad económica que debe ser incluido en la carpeta:

- Debe presentar un detalle de las tareas a realizar a lo largo del tiempo con fecha límite de entrega en la última semana de cursada de Trabajo de Diploma.
- Debe contener el detalle de las horas dedicadas a cada tarea, los recursos humanos afectados, el costo horario de cada recurso y el costo total del proyecto desde el relevamiento hasta la implementación.
- Debe contener detalle de todo aquel hardware y software que se deba instalar para hacer este proyecto viable, con sus costos.
- Debe contener los costos de capacitación de los empleados.

Considerando el aspecto financiero, en este estudio se plantean los distintos escenarios de flujo de fondos (entrantes y salientes), así como el origen de los recursos. Para ello se calculará el Valor Actual Neto del proyecto y la Tasa Interna de retorno del mismo.

4.6.4.1. Valor Actual Neto (VAN)

Toda evaluación de un proyecto requiere comparar el costo de inversión (generalmente realizado al inicio del proyecto) con las entradas de efectivo netas que ocurren en períodos sucesivos de tiempo. Pero debemos tener en cuenta que el dinero se devalúa. No tiene el mismo valor hoy que el mes próximo y ni mucho menos en períodos posteriores.

El Valor Actual Neto representa cuanto vale hoy el dinero que recibiré en el futuro.

Para poder calcularlo, el dinero que se recibirá a futuro deberá ser descontado según una tasa porcentual apropiada, que por lo regular es la tasa de interés vigente.

Así, el Valor Actual Neto (VAN) de un determinado monto de dinero (M) que se percibirá en un determinado período (n) en un contexto con una tasa de interés fija por período (i) está representado por la fórmula:

$$VA = \frac{M}{(1+i)^n} \quad (I)$$

Si tuviésemos ingresos en los períodos 1 ($n=1$), 2 ($n=2$) y 3 ($n=3$), tendríamos que repetir la ecuación (I) para cada uno de ellos y sumar los resultados obtenidos. Así, si suponemos que recibiré dinero (M_n) correspondientes a cada uno de los 3 períodos, en un mercado con una tasa de interés por periodo (i), tendríamos que:

$$VA = \frac{M_1}{(1+i)^1} + \frac{M_2}{(1+i)^2} + \frac{M_3}{(1+i)^3}$$

Para el Valor Actual Neto del término n -simo,

$$VA = M_0 + \frac{M_1}{(1+i)^1} + \frac{M_2}{(1+i)^2} + \frac{M_3}{(1+i)^3} + \dots + \frac{M_n}{(1+i)^n}$$

Lo que es igual a:

$$VA = \sum_{j=0}^n \frac{M_j}{(1+i)^j} \quad (II)$$

4.6.4.2. Costo Neto (CT)

De la misma manera que debo descontar el dinero que percibiré a futuro de acuerdo a una determinada tasa de interés, también debo descontar los importes de las erogaciones que debo hacer a lo largo del desarrollo.

Sabemos que el costo total del proyecto podemos asumirlo en un solo periodo al inicio (periodo =0) o en uno o más periodos (periodo = {0,1,2,...,n}).

En todos los casos la fórmula es similar a la de Valor Actual Neto, salvo que reemplazamos M (monto) por C (costo).

$$CT = C_0 + \frac{C_1}{(1+i)^1} + \frac{C_2}{(1+i)^2} + \frac{C_3}{(1+i)^3} + \dots + \frac{C_n}{(1+i)^n} \quad \text{Dónde CT = Costo Total}$$

O, lo que igual a:

$$CT = \sum_{j=0}^n \frac{C_j}{(1+i)^j} \quad (III)$$

4.6.4.3. Valor Actual Neto (VAN)

El Valor Actual Neto (VAN) de un proyecto se calcula deduciendo del Valor Actual Neto (VAN) los costos del proyecto.

$$VAN = VA - CT$$

Si suponemos que la salida de fondos se produce toda de una sola vez y en el periodo inicial, entonces $CT=C_0$ y el VAN es igual a :

$$VAN = \left(\sum_{j=0}^n \frac{M_j}{(1+i)^j} \right) - C_o$$

En cambio, si los egresos se distribuyen en los periodos, entonces restamos las ecuaciones II y III, quedando la siguiente fórmula para el VAN

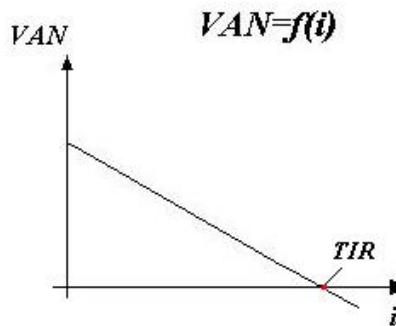
$$VAN = \left(\sum_{j=0}^n \frac{M_j}{(1+i)^j} \right) - \left(\sum_{j=0}^n \frac{C_j}{(1+i)^j} \right)$$

Agrupando los términos:

$$VAN = \left(\sum_{j=0}^n \frac{M_j - C_j}{(1+i)^j} \right)$$

4.6.4.4. Tasa Interna de Retorno (TIR) y Rendimiento del proyecto

Como vemos, el VAN es inversamente proporcional a la tasa de interés. Es decir que a medida que la tasa de interés aumenta, el Valor Actual Neto (VAN) de mi proyecto, disminuye hasta que el VAN se hace cero. Si representáramos gráficamente el VAN en función de la tasa de interés obtendríamos un gráfico similar al de la figura siguiente:



En el punto en el cual la función corta al eje de las abscisas (i), el VAN es igual a cero. El valor de la tasa i en ese punto representa la tasa interna de retorno (TIR).

Como definición podemos decir que: "La tasa interna de retorno es aquella que hace que el VAN del proyecto sea igual a cero" o, en el mismo sentido, "aquella para la cual el Valor Actual Neto (VAN) del proyecto es igual a los costos"

Para saber si un proyecto es reddituable o no, se debe comparar el TIR con la tasa de mercado vigente.

A la diferencia entre el TIR y la tasa de mercado se la denomina **Rendimiento del proyecto** y su fórmula es:

$$R = TIR - Tc$$

Donde: TIR es la tasa interna de retorno
Tc es la tasa de mercado.

Si: $R > 0$ El proyecto es viable
 $R = 0$ El proyecto no da pérdidas ni ganancias
 $R < 0$ El proyecto no es viable.

De este análisis podemos llegar a la conclusión de que no siempre un valor de $TIR < 0$ indica que el proyecto no es viable. Por ejemplo, supongamos un proyecto que se desarrolla en un contexto donde hay deflación, con una tasa de interés de -5% mensual, y un TIR de -2% mensual. El rendimiento del proyecto será de:

$$R = TIR - Tc = (-2\%) - (-5\%) = -2\% + 5\% = +3\%$$

Vemos que a pesar de ser el TIR negativo, el proyecto es viable en ese contexto ya que nos proporciona un rendimiento de + 3%.

Se deben plantear por lo menos tres escenarios distintos para el flujo de fondos.

Características del estudio de viabilidad financiera que debe ser incluido en la carpeta:

- Al momento de ser aprobada la carpeta de proyecto deberá contar con, al menos, tres escenarios distintos para el flujo de fondos presentando en cada caso los cálculos correspondientes al VAN. Para el cálculo deberá considerarse una tasa de interés similar a la vigente en el mercado para préstamos personales.
- Cada escenario deberá ser presentado en hoja aparte.
- De estos tres cálculos, el cliente optará por uno, el cual será utilizado en la propuesta de trabajo y el presupuesto.

NOTA: Si bien los escenarios son distintos, todos se realizan para un mismo período. Por lo tanto, la tasa de interés que se utilice debe ser la misma en todos los casos. Cada escenario deberá ser presentado en hoja separada de los otros.

4.6.5. Viabilidad de Gestión.

Cátedra	Carácter
TC1	Opcional

El objetivo de este estudio es, principalmente, definir si existen las condiciones mínimas necesarias para garantizar la viabilidad de la implementación, tanto en lo estructural como en lo funcional.

El estudio debe demostrar que existen las capacidades gerenciales para llevar a cabo el proyecto en forma eficiente. Una de varias opciones para medir esto se relaciona directamente con la calidad del trabajo realizado por el evaluador del proyecto de software. Si el estudio de viabilidad económica exhibe deficiencias notorias, es muy posible que se presuma que la incapacidad para hacer un buen análisis o para hacerse asesorar en una etapa tan decisiva del proyecto, se mantendrán una vez implementado el proyecto.

4.6.6. Viabilidad Comercial

Cátedra	Carácter
TC1	Opcional

Indica si el mercado es o no sensible al software producido por el proyecto y la aceptabilidad que tendría en su consumo o uso, permitiendo de esta forma, determinar la postergación o rechazo

PROYECTO FINAL TC1/TD

de un proyecto sin tener que asumir los costos que implica un estudio económico completo. Tiene por objeto definir la cuantía de la demanda, ingresos de operación y las inversiones implícitas.

4.7. Glosario

Cátedra	Carácter
ASA	Requerido

Es un documento que define los principales términos usados en el proyecto. Permite establecer una terminología consensuada.

Se puede visualizar un ejemplo de este artefacto en la sección [Documento Glosario](#) del proyecto modelo.

4.8. Modelo de Casos de Uso

Cátedra	Carácter
ASA (80%)/TC1 (20%)	Requerido

El modelo de Casos de Uso presenta las funciones del sistema y los actores que hacen uso de ellas. Se representa mediante Diagramas de Casos de Uso.

Se puede visualizar un ejemplo de este artefacto en el diagrama de Casos de Uso de ["Gestión de Ventas"](#).

4.9. Especificaciones de Casos de Uso.

Cátedra	Carácter
ASA (80%)/TC1 (20%)	Requerido

Para los casos de uso que lo requieran (cuya funcionalidad no sea evidente o que no baste con una simple descripción narrativa) se realiza una descripción detallada utilizando una plantilla de documento, donde se incluyen: precondiciones, post-condiciones, flujo de eventos, requisitos no-funcionales asociados. También, para casos de uso cuyo flujo de eventos sea complejo podrá adjuntarse una representación gráfica mediante un Diagrama de Actividad.

Se puede visualizar un ejemplo de este artefacto en la [Especificación del caso de uso "elaborar pedido"](#).

4.10. Especificaciones Adicionales

Cátedra	Carácter
ASA/TC1	Requerido

Este documento capturará todos los requisitos que no han sido incluidos como parte de los casos de uso y se refieren requisitos no-funcionales globales. Dichos requisitos incluyen: requisitos legales o normas, aplicación de estándares, requisitos de calidad del producto, tales como: confiabilidad, desempeño, etc., u otros requisitos de ambiente, tales como: sistema operativo, requisitos de compatibilidad, etc.

4.11. Mapa de Navegación

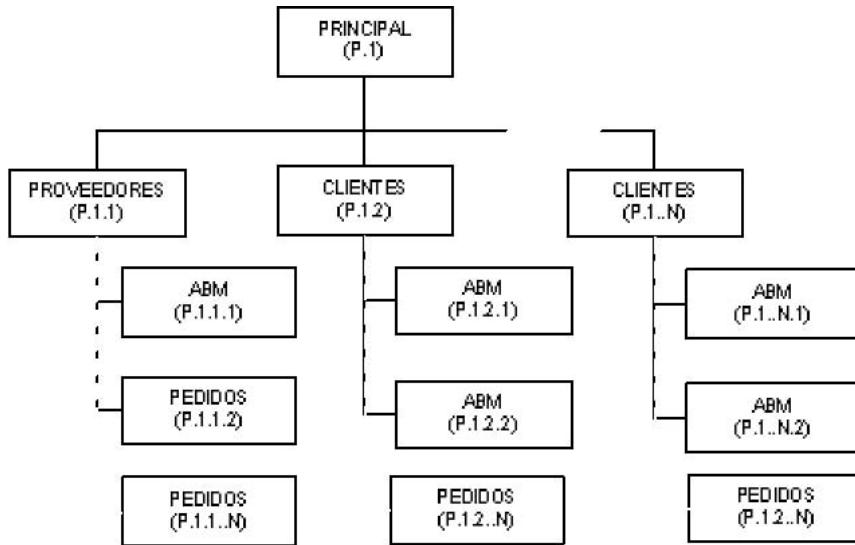
Cátedra	Carácter
TC1	Requerido

En el mapa de navegación deben estar representadas todas las pantallas del sistema con las correspondientes rutas de acceso a ellas.

Es importante notar la nomenclatura tanto en amplitud como en profundidad. Cada pantalla lleva un nombre y un código que representa su ubicación en el mapa.

PROYECTO FINAL TC1/TD

A continuación se presenta un ejemplo.



Se deben mantener conceptos de ergonomía y usabilidad para el diseño de las pantallas.

4.12. Prototipos de Interfaces de Usuario

Cátedra	Carácter
TC1	Requerido

Se trata de prototipos que permiten al usuario hacerse una idea más o menos precisa de las interfaces que proveerá el sistema y así, conseguir retroalimentación de su parte respecto a los requisitos del sistema. Estos prototipos se realizarán como: dibujos a mano en papel, dibujos con alguna herramienta gráfica o prototipos ejecutables interactivos, siguiendo ese orden de acuerdo al avance del proyecto. Sólo los de este último tipo serán entregados al final de la fase de Elaboración, los otros serán desechados. Asimismo, este artefacto, será desecharido en la fase de Construcción en la medida que el resultado de las iteraciones vayan desarrollando el producto final.

Se puede visualizar un ejemplo de este artefacto en la sección [Prototipos de Interfaces de Usuario](#) del proyecto modelo.

4.13. Diagrama de Clases

Cátedra	Carácter
ASA	Requerido

Este modelo establece la realización de los casos de uso en clases y pasando desde una representación en términos de análisis (sin incluir aspectos de implementación) hacia una de diseño (incluyendo una orientación hacia el entorno de implementación), de acuerdo al avance del proyecto.

La asignación de responsabilidades a las clases deberá seguir el criterio explicado a continuación.

Las responsabilidades se relacionan con las obligaciones de un objeto respecto de su comportamiento.

La responsabilidad no es lo mismo que un método, pero los métodos se implementan para llevar a cabo las responsabilidades. Estas responsabilidades pertenecen, esencialmente, a dos categorías: **hacer** y **conocer**.

Entre las responsabilidades de un objeto relacionadas con el hacer se encuentran:

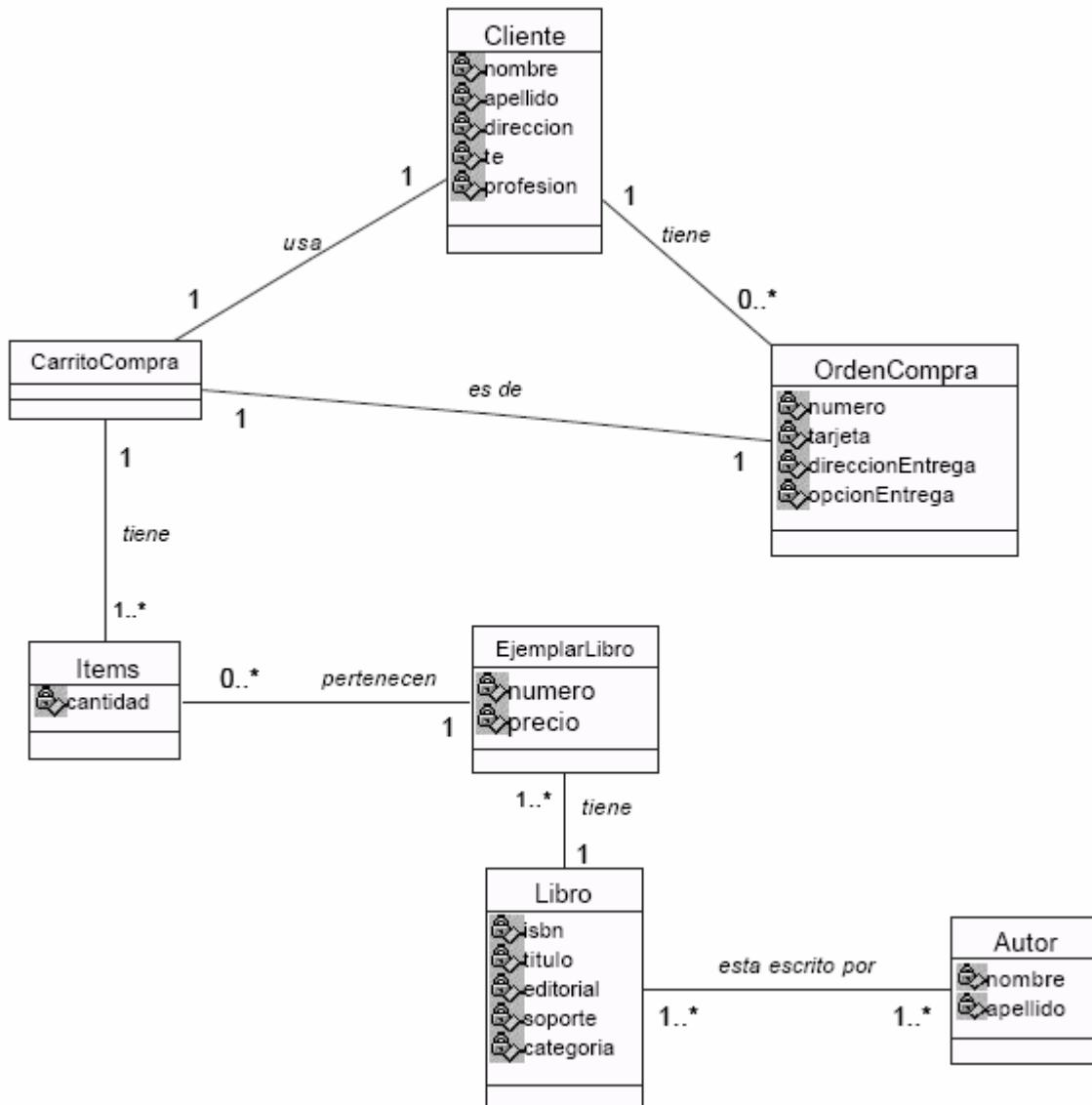
- Hacer algo uno mismo.
- Iniciar una acción en otros objetos.
- Controlar y coordinar actividades en otros objetos.

Entre las responsabilidades de un objeto relacionadas con el conocer se encuentran:

- Conocer los datos privados encapsulados.
- Conocer los objetos relacionados.
- Conocer las cosas que se pueden derivar o calcular.

"las responsabilidades relevantes de **conocer** a menudo se pueden inferir a partir del modelo del dominio"

Un ejemplo:



El Patrón “Experto” [Larman]

Problema: ¿Cuál es el principio fundamental en virtud del cual se asignan las responsabilidades en el diseño orientado a objetos?

PROYECTO FINAL TC1/TD

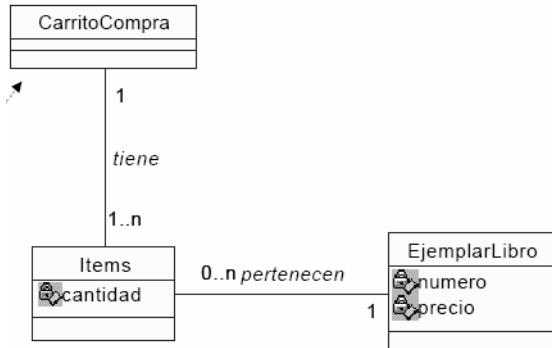
Solución: Asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.

Ejemplo: ¿quién es el responsable de conocer el total de una venta?

Desde el punto de vista del patrón Experto, deberíamos buscar la clase de objetos que posee información sobre los Items vendidos.

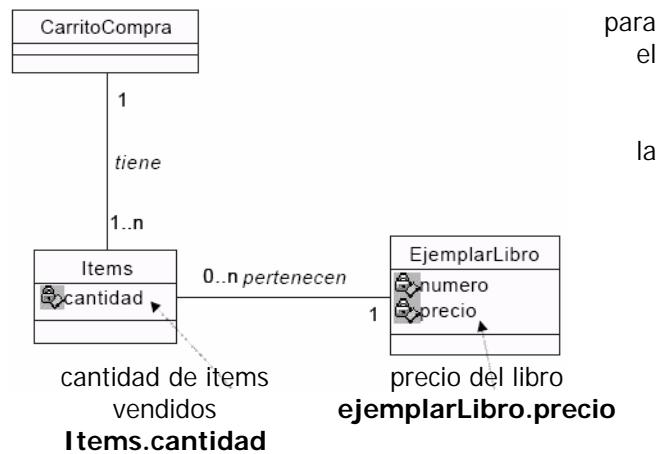
El objeto que conoce esto es CarritoCompra.

Nota: buscamos inicialmente en las clases del modelo de dominio.

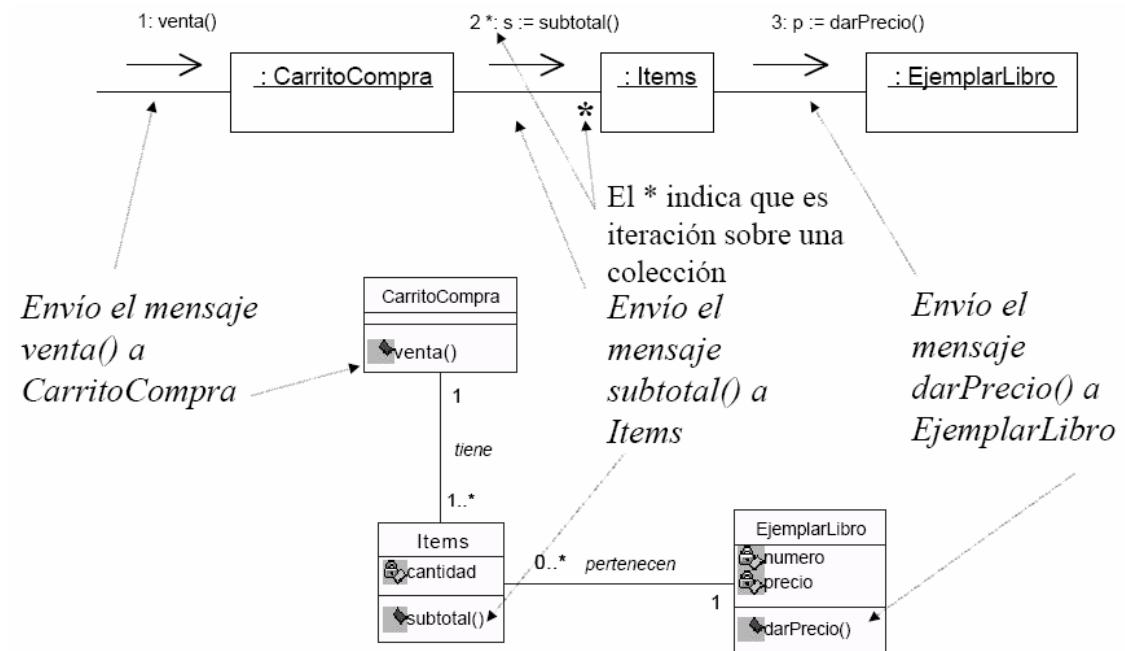


¿Qué información hace falta saber determinar la cantidad de Items vendidos y precio para saber la venta total?

La cantidad de Ítems vendidos está en clase Items y el precio, en EjemplarLibro, ambos tienen la información necesaria para realizar la responsabilidad.

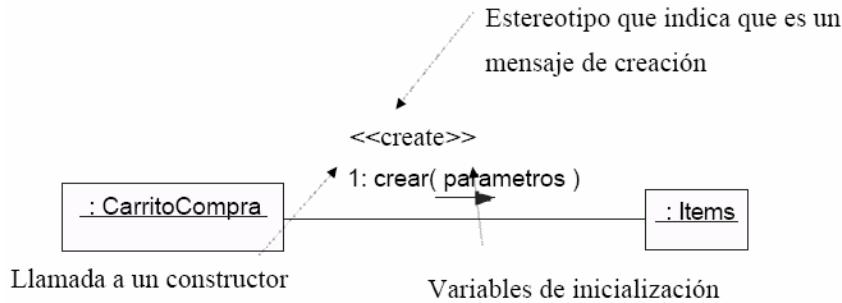


Seguimos con el ejemplo:



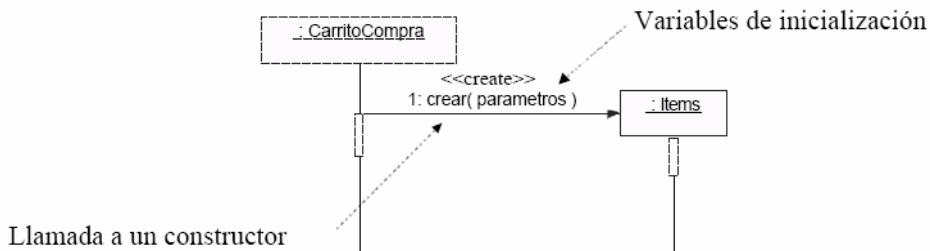
El Patrón “Creador” [Larman]

El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El objetivo de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento.



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

La creación de objetos es una de las actividades más frecuentes en un sistema orientado a objetos. En consecuencia, conviene contar con un principio general para asignar las responsabilidades concernientes a ella.



Solución: Asignarle a la clase B la responsabilidad de crear una instancia de la clase A en uno de los siguientes casos:

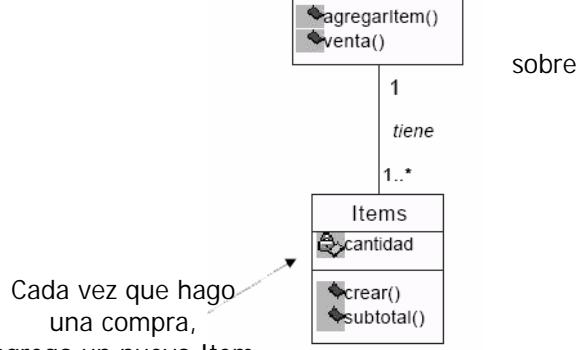
- B agrega los objetos de A.
- B contiene los objetos de A.
- B registra las instancias de los objetos de A.
- B tiene los datos de inicialización que serán enviados a A cuando este objeto sea creado (B es un experto respecto a la creación de A).

Beneficios: Se brinda apoyo a un bajo acoplamiento, lo cual supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización.

Ejemplo: En la aplicación ¿quién debería encargarse de crear una instancia de items?

Desde el punto de vista del patrón Creador, deberíamos buscar una clase que agregue, contenga, y realice otras operaciones este tipo de instancias.

CarritoCompra
contiene uno o
más Items



El Patrón “Bajo acoplamiento”

El acoplamiento es la medida de la fuerza con que un elemento está conectado con otros.

Un alto acoplamiento implica que:

- Los cambios en las clases relacionadas fuerzan cambios en las clases locales.
- Son difíciles de entender de manera aislada
- Son difíciles de reutilizar, debido a que su uso requiere la presencia adicional de las clases de las que depende

Problema: ¿cómo soportar bajas dependencias, bajo impacto en el cambio e incremento en la reutilización?

Solución: asignar las responsabilidades a las clases de manera de mantener el acoplamiento bajo

Beneficios: las clases son más independientes, lo que reduce el impacto al cambio

El Patrón “Alta cohesión”

La cohesión es una medida de la especificidad de una responsabilidad. Una clase con baja cohesión hace muchas cosas no relacionadas y adolece de los siguientes problemas: Difíciles de entender, reutilizar y mantener.

Regla empírica: una clase con alta cohesión tiene un número relativamente pequeño de métodos con funcionalidad altamente relacionada.

Problema: ¿cómo mantener la complejidad manejable?

Solución: asignar las responsabilidades de manera que la cohesión permanezca alta.

Beneficios: se incrementa la claridad, se simplifica el mantenimiento y las mejoras, implica generalmente bajo acoplamiento

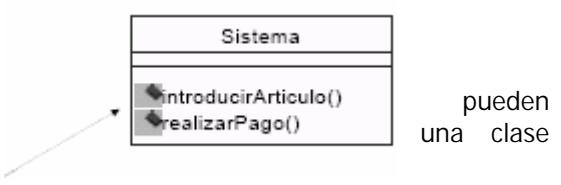
El Patrón “Controlador”

Problema: ¿Quién debe ser el responsable de gestionar un evento de entrada al sistema? (Un evento del sistema de entrada es un evento generado por un actor externo)

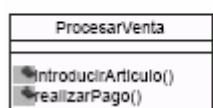
Solución: asignar la responsabilidad de recibir o manejar un mensaje de evento del sistema a una clase que:

- a) represente el sistema global
- b) represente un escenario de caso de uso donde tiene lugar el evento

Durante el análisis, las operaciones del sistema asignarse a la clase “Sistema”, eso no significa que software “Sistema” las lleve a cabo.

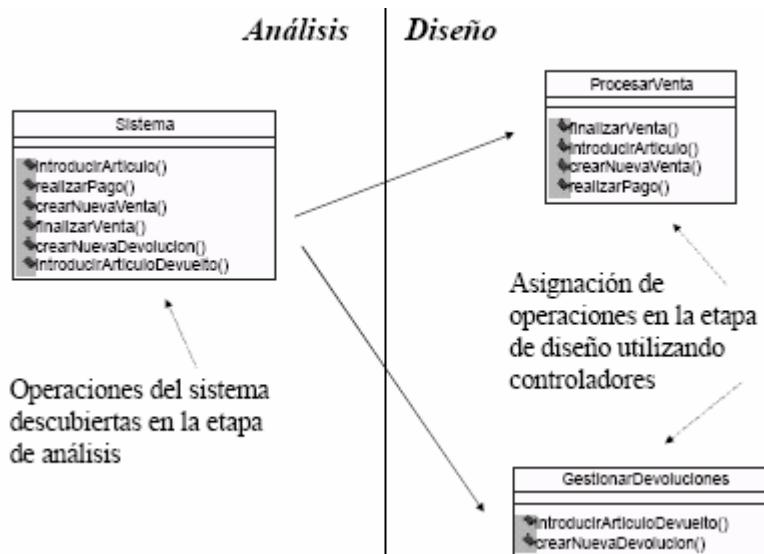


Durante el diseño, se le asignan las responsabilidades de las operaciones del sistema a una clase controlador (la clase controlador no es una clase del dominio de la aplicación)

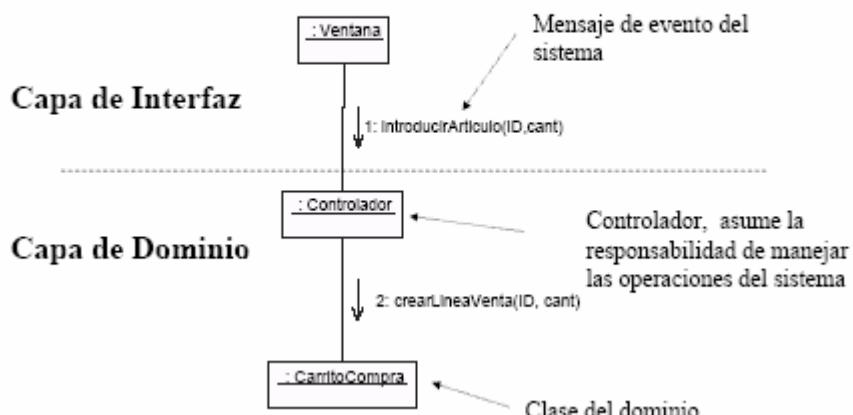


- Un objeto controlador no pertenece a la capa de interfaz.
- Generalmente no hace trabajo, lo delega a otros objetos.
- El controlador es una especie de fachada sobre la capa de dominio para la capa de interfaz.
- Durante el diseño se asignan a una o más clases controlador las operaciones del sistema que se identifican durante el análisis.
 - Normalmente se utiliza una misma clase controlador para los eventos del sistema de un caso de uso.
 - Si no hay muchos eventos al sistema puedo usar una única clase controlador de fachada.

Un ejemplo:



La capa de interfaz no maneja eventos del sistema

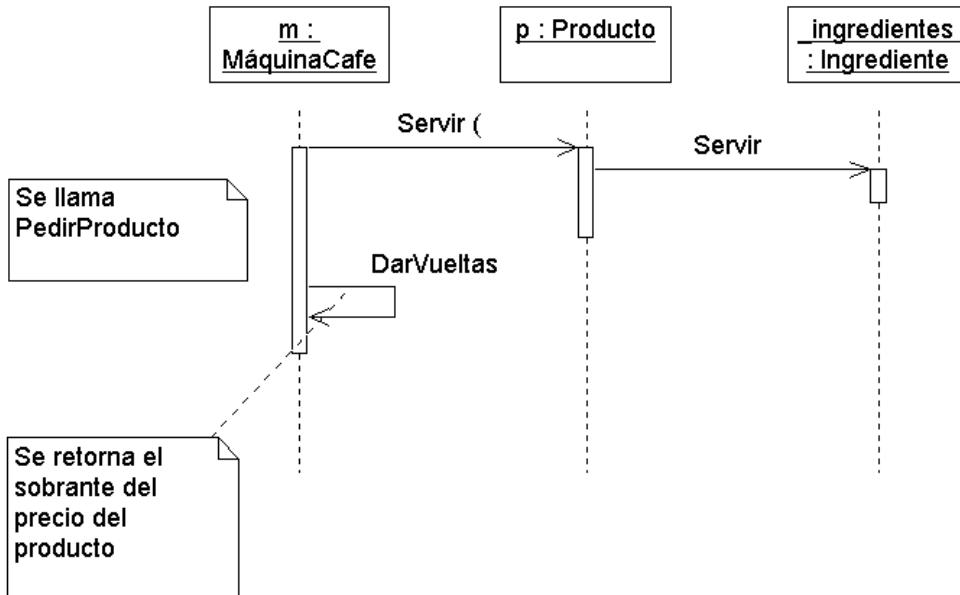


4.14. Diagrama de Secuencia

Cátedra	Carácter
ASA	Requerido

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo. Esta descripción es importante porque puede dar detalle a los casos de uso, aclarándolos al nivel de mensajes de los objetos existentes, como también muestra el uso de los mensajes de las clases diseñadas en el contexto de una operación.

A continuación podemos ver un ejemplo de este tipo de diagramas en el cual tenemos instancias de tres clases. El objeto m es una instancia de la clase MáquinaCafé, el objeto p es una instancia de la clase producto y el objeto ingredientes es una instancia del objeto ingrediente.



Un diagrama de secuencia está compuesto por:

- Línea de Vida de un objeto: representada por una línea vertical punteada con un rectángulo de encabezado y con rectángulos a través de la línea principal que denotan la ejecución de métodos (véase activación). El rectángulo de encabezado contiene el nombre del objeto y el de su clase, en un formato nombreObjeto: nombreClase. Por ejemplo, el objeto m, instancia de la clase MáquinaCafé envía dos mensajes seguidos para dar respuesta a la operación PedidoProducto: Servir al objeto p de la clase Producto y DarVueltas a sí mismo.
- Activación: Muestra el periodo de tiempo en el cual el objeto se encuentra desarrollando alguna operación, bien sea por sí mismo o por medio de delegación a alguno de sus atributos. Se denota como un rectángulo delgado sobre la línea de vida del objeto. En el ejemplo anterior el objeto ingredientes se encuentra activado mientras ejecuta el método correspondiente al mensaje Servir; el objeto p se encuentra activo mientras se ejecuta su método Servir (que ejecuta ingredientes.Servir) y el objeto m se encuentra activo mientras se ejecuta p.Servir y DarVueltas.
- Mensaje: El envío de mensajes entre objetos se denota mediante una línea sólida dirigida, desde el objeto que emite el mensaje hacia el objeto que lo ejecuta. En el ejemplo anterior el objeto m envía el mensaje Servir al objeto p y un poco más adelante en el tiempo el objeto m se envía a sí mismo el mensaje DarVueltas.

4.15. Diagrama de Componentes

Cátedra	Carácter
TC1	Requerido

Este modelo es una colección de componentes y los subsistemas que los contienen. Estos componentes incluyen: archivos ejecutables, archivos de código fuente, y todo otro tipo de archivos necesarios para la implantación y despliegue del sistema. (Este modelo es sólo una versión preliminar al final de la fase de Elaboración, posteriormente tiene bastante refinamiento).

Se puede visualizar un ejemplo de este artefacto en la sección [Diagrama de Componentes](#) del proyecto modelo.

4.16. DER

Cátedra	Carácter
ASA	Requerido

Previendo que la persistencia de la información del sistema será soportada por una base de datos relacional, este modelo describe la representación lógica de los datos persistentes, de acuerdo con el enfoque para modelado relacional de datos. Para expresar este modelo se utiliza un Diagrama de Entidad Relación (DER) u opcionalmente un Diagrama de Clases (donde se utiliza un profile UML para Modelado de Datos, para conseguir la representación de tablas, claves, etc.).

El modelo de datos debe dar respuesta a preguntas específicas que son importantes para cualquier aplicación de procesamiento de datos.

Para poder dar esas respuestas el modelado de datos utiliza el diagrama entidad-relación (DER), el cual permite identificar objetos de datos (entidades) y las relaciones que las unen, mediante una notación gráfica.

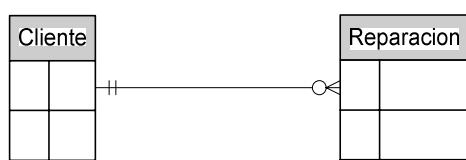
La representación del modelo de datos se compone de tres partes relacionadas: las entidades, los atributos que las describen y las relaciones que las conecta entre sí.

- Una **entidad** es una representación de un conjunto de atributos que definen a un objeto de datos.
- Un **atributo** es definido como las propiedades de objeto o entidad.
- Las **relaciones** Permiten relacionar distintas entidades a partir de sus atributos claves. Para poder clarificar qué son las relaciones debemos introducir dos conceptos nuevos: la cardinalidad y la modalidad.
- La **cardinalidad** indica el número máximo de relaciones de objeto que puede participar en una relación.
- La **modalidad** indica el número mínimo de ocurrencias. Es "cero" (o "ninguna") si no hay necesidad de ocurrencia obligatoria. La modalidad es "uno" (o "una") si la ocurrencia de la relación es obligatoria.

Representación	Modalidad	Cardinalidad	Ocurrencia
—○+	Ninguna	una	Ninguna o una
—++	Una	Una	Una o Una
—○	Ninguna	Muchas	Ninguna o muchas
—	Una	Muchas	Una o muchas

Sea el ejemplo siguiente en el que tenemos una entidad CLIENTE y otra entidad REPARACIÓN. Vemos que del lado Cliente la modalidad es uno y la cardinalidad es uno.

En cambio del lado Reparacion, la modalidad es cero y la cardinalidad es uno.



La pregunta es ¿Cómo se lee esta relación?

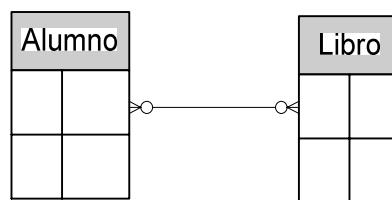
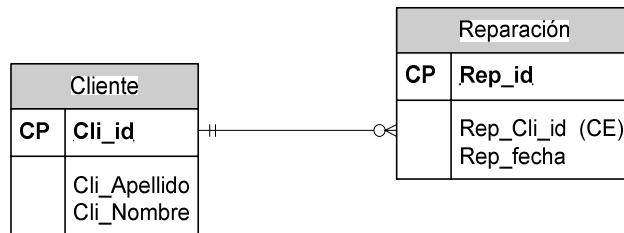
"A cada cliente le corresponden ninguna o muchas reparaciones y a cada reparación corresponde a uno y sólo un cliente"

PROYECTO FINAL TC1/TD

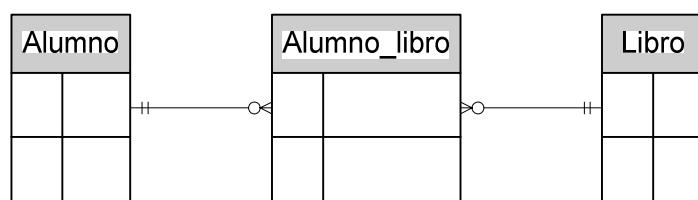
Para poder establecer las relaciones entre las entidades deben existir claves primarias en las entidades, formadas por uno o más atributos, que permitan identificar únicamente las tuplas de cada una de ellas.

Las claves foráneas son referencias de una entidad hacia otra. El o los atributos que referencien a otra entidad deben hacerlo a la clave primaria completa y deben tener el mismo dominio que en la entidad referenciada.

En el ejemplo siguiente vemos que la entidad Cliente tiene como clave primaria al atributo Cli_id, y dos atributos Cli_Apellido y Cli_Nombre. La entidad Reparación tiene como clave primaria al atributo Rep_id y dos atributos Rep_Cli_id y Rep_Fecha.



Para resolver este impedimento, creamos una entidad de asociación en la que se registran todas las relaciones entre los alumnos y los libros



De esta manera, nos quedan dos relaciones uno a muchos de las entidades originales (alumno y libro) con la entidad de asociación (Alumno_libro).

PROYECTO FINAL TC1/TD

4.16.1. Restricciones del modelo de datos

Un esquema de base de datos es un conjunto de relaciones con un conjunto de restricciones de integridad.

El modelo relacional de datos posee 4 tipos de restricciones:

•**De dominio:**

"El valor del atributo debe pertenecer al dominio o, en caso de estar permitido, ser nulo "

Por ejemplo:

El valor de todos los atributos de las entidades debe pertenecer a los dominios que se hayan definido para cada uno de ellos.

•**De clave:**

"En una entidad, no puede haber dos tuplas con el mismo valor de clave primaria".

Por ejemplo:

En la entidad Cliente, el atributo *Cli_id* no puede tener el mismo valor en dos tuplas diferentes.

•**De Entidad:**

"No puede haber valores nulos en la clave primaria de una tupla"

Ninguna tupla de una entidad puede tener con valor nulo a alguno de los atributos que forman su clave primaria.

Por ejemplo:

En la entidad Cliente, el atributo *Cli_id* no puede tener valor nulo

•**De Integridad referencial:**

"Cuando un atributo es clave foránea de una entidad, debe mantener valores concordantes con la clave primaria de la entidad con la que se relaciona"

Por ejemplo:

En la entidad Reparación, el atributo *Rep_cli_id* referencia al atributo *Cli_id* de la entidad Cliente. Para no violar la restricción de integridad referencial el dominio de *Rep_cli_id* debe ser el mismo que el de *Cli_id* y el valor contenido en *Rep_cli_id* debe aparecer como valor en *Cli_id* o ser nulo.

4.16.2. Normalización de las entidades de las bases de datos

Las entidades que formen parte de los trabajos de campo deben estar normalizadas de acuerdo a las 3 primeras formas normales, las que se recuerdan a continuación:

Primera forma normal (1FN)

"Los atributos que forman parte de una entidad deben tomar sus valores de un conjunto de valores atómicos"

Un valor atómico es aquel que si se lo divide, las partes resultantes no tienen significado propio. El error más común al momento de modelar una entidad de datos en la que haya que establecer un domicilio es el de colocar en un mismo atributo Dirección los datos de la calle y número. Lo correcto es establecer dos atributos, uno para la calle, otro para el número y otro para la localidad.

Incorrecto	Correcto
\$24280	\$24280

The diagram illustrates the normalization of a value. On the left, under the heading "Incorrecto", there is a large black dollar sign (\$) followed by the digits 24280. A horizontal line extends from the right side of this value. On the right, under the heading "Correcto", there is a large black dollar sign (\$) followed by a comma (,), then the digits 24,280. Below the "Incorrecto" section, the text "Página 19 de 102" is visible.

Segunda forma normal (2FN)

"Todo atributo no clave de depender funcionalmente en forma completa de la clave"

De esto podemos deducir que si una entidad cumple con la primera forma normal y tiene clave primaria formada por un solo atributo, entonces está en segunda forma normal (2FN).

Cuando la clave primaria la forma más de un atributo, no puede haber un atributo no clave que dependa de alguno de los atributos claves.

Tercera forma normal (3FN)

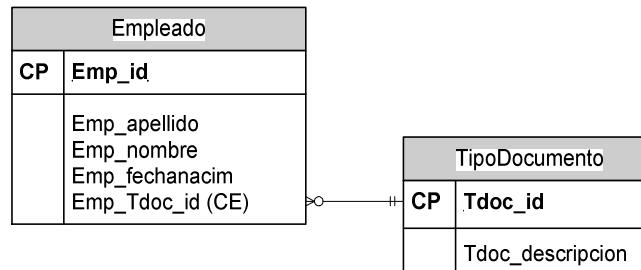
"Todos los atributos no clave no tienen dependencia funcional entre ellos".

Esto significa que los atributos no clave no pueden depender de otro atributo no clave.

Si se presenta esa condición se debe crear una nueva entidad a la cual referenciar como clave externa.

En la tabla de la derecha vemos que existe un atributo Emp_TipoVehiculo y un atributo Emp_RuedasVehic, que depende funcionalmente del tipo de vehículo. Así si el tipo de vehículo es un auto, la cantidad de ruedas es 4. Y si el tipo de vehículo es una bicicleta, entonces el número de ruedas del vehículo es 2.

Empleado	
CP	Emp_id
	Emp_Calle
	Emp_numero
	Emp_Localidad
	Emp_TipoVehiculo
	Emp_RuedasVehic



PROYECTO FINAL TC1/TD

Entidad	Mnem	Descripción		
EMPLEADO	emp_	Datos personales del empleado		
Atributos				
Nombre	Descripción	Tipo	Tamaño	Dominio
emp_id	Código identificación	Numérico	10	0-9
emp_nombre	Apellido del profesional	Texto	30	A – Z, Ç,'
...

Cuando se definen los dominios se debe tener mucho cuidado con los tipos de datos que se incorporan a él. Se debe tener en cuenta que, por ejemplo, en un email puede haber números, si se aceptan los puntos o guiones para separar otros caracteres, etc.

4.16.4. Características de los diagramas de entidad-relación que deben ser incluidos en el proyecto.

Al momento de ser aprobada la carpeta de proyecto deberá contar con, al menos, dos diagramas de Entidad Relación (DER):

- El correspondiente al objeto del negocio: tendrá entidades propias de él, por lo tanto cada alumno establecerá las entidades y atributos que considere suficientes para satisfacer las necesidades de la aplicación.
- El correspondiente al módulo de seguridad: Donde se modelarán los datos que permitan realizar las operaciones de Login, administración de perfiles de usuario (utilizando el método de usuario, familia, patente), manejo de múltiples idiomas, bitácora de registro de actividades en el programa.

Ambos diagramas deben cumplir con los siguientes requisitos:

- Los nombres de las entidades deben estar en singular (Por ejemplo: USUARIO)
- Todas las entidades incluidas deben estar en la tercera forma normal (3FN).
- Todas relaciones entre tablas deben tener indicadas la modalidad y la cardinalidad en cada uno de los extremos
- Los atributos que son parte de la clave principal deben ser identificados mediante las siglas CP o PK.
- Los atributos que son claves externas deben ser identificados con las siglas CE (si se usó CP en las claves primarias o FK, si se usó PK en las claves primarias).
- Los nombres de las claves primarias deben ser encabezados por el mnemónico correspondiente a la entidad a la que pertenece. En la figura 5.5, la entidad EMPLEADO tiene como mnemónico a emp_. Entonces, el nombre de todos los atributos deben ser precedidos por emp_ como en el caso emp_id, emp_nombre. El uso de esta nomenclatura permite una rápida identificación de la relación entidad-atributo.
- Los nombres de las claves foráneas deben estar formados por el mnemónico de la entidad a la que pertenece seguido por el nombre completo del atributo clave al que hace referencia. En la figura 5.5 podemos ver que la entidad EMPLEADO se relaciona con la entidad TIPODOCUMENTO a través del atributo emp_tdoc_id. El uso de esta nomenclatura permite identificar rápidamente las relaciones entre entidades.

En todas las entidades del diagrama entidad-relación de seguridad se deben contemplar los dígitos verificadores horizontales y verticales.

Se puede visualizar un ejemplo de este artefacto en la sección [Modelo de Datos](#) del proyecto modelo.

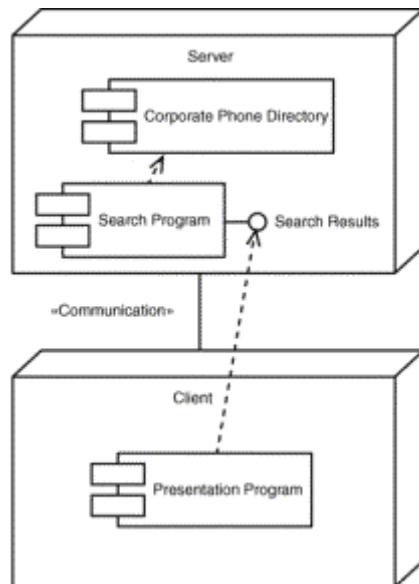
4.17. Modelo de Despliegue

PROYECTO FINAL TC1/TD

Cátedra	Carácter
TC1	Opcional

Este modelo muestra el despliegue la configuración de tipos de nodos del sistema, en los cuales se hará el despliegue de los componentes.

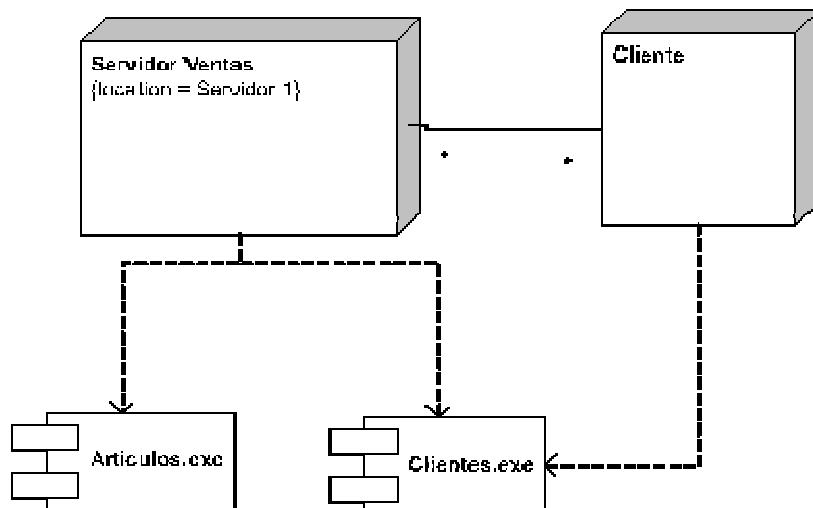
En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados. Es decir se sitúa el software en el hardware que lo contiene. Cada Hardware se representa como un nodo.



Un nodo se representa como un cubo, un nodo es un elemento donde se ejecutan los componentes, representan el despliegue físico de estos componentes.

Aquí tenemos dos nodos, el cliente y el servidor, cada uno de ellos contiene componentes. El componente del cliente utiliza una interfaz de uno de los componentes del servidor. Se muestra la relación existente entre los dos Nodos. Esta relación podríamos asociarle un estereotipo para indicar que tipo de conexión disponemos entre el cliente y el servidor, así como modificar su cardinalidad, para indicar que soportamos diversos clientes.

Como los componentes pueden residir en mas de un nodo podemos situar el componente de forma independiente, sin que pertenezca a ningún nodo, y relacionarlo con los nodos en los que se sitúa.



PROYECTO FINAL TC1/TD

4.17.1. Características de los diagramas de despliegue que se deben incluir en la carpeta de proyecto:

- Deben tener correspondencia uno a uno con los objetos y clases empleados en los diagramas de secuencia.
- Las interfaces deben estar bien definidas.

Se puede visualizar un ejemplo de este artefacto en la sección [Diagrama de Despliegue](#) del proyecto modelo.

4.18. Especificación de Casos de Prueba

Cátedra	Carácter
TC1	Opcional

Cada prueba es especificada mediante un documento que establece las condiciones de ejecución, las entradas de la prueba, y los resultados esperados. Estos casos de prueba son aplicados como pruebas de regresión en cada iteración. Cada caso de prueba llevará asociado un procedimiento de prueba con las instrucciones para realizar la prueba, y dependiendo del tipo de prueba dicho procedimiento podrá ser automatizable mediante un script de prueba.

Se puede visualizar un ejemplo de este artefacto en la [Especificación de Caso de Prueba "Consultar pedidos no atendidos"](#).

4.19. Solicitud de Cambio

Cátedra	Carácter
ASA/TC1	Opcional

Los cambios propuestos para los artefactos se formalizan mediante este documento. Mediante este documento se hace un seguimiento de los defectos detectados, solicitud de mejoras o cambios en los requisitos del producto. Así se provee un registro de decisiones de cambios, de su evaluación e impacto, y se asegura que éstos sean conocidos por el equipo de desarrollo. Los cambios se establecen respecto de la última *línea base* (el estado del conjunto de los artefactos en un momento determinado del proyecto) establecida. En nuestro caso al final de cada iteración se establecerá una *línea base*.

4.20. Plan de Iteración

Cátedra	Carácter
ASA	Opcional

Es un conjunto de actividades y tareas ordenadas temporalmente, con recursos asignados, dependencias entre ellas. Se realiza para cada iteración, y para todas las fases.

4.21. Evaluación de Iteración

Cátedra	Carácter
ASA	Opcional

Este documento incluye la evaluación de los resultados de cada iteración, el grado en el cual se han conseguido los objetivos de la iteración, las lecciones aprendidas y los cambios a ser realizados.

4.22. Lista de Riesgos

Cátedra	Carácter
TC1	Opcional

Este documento incluye una lista de los riesgos conocidos y vigentes en el proyecto, ordenados en orden decreciente de importancia y con acciones específicas de contingencia o para su mitigación.

PROYECTO FINAL TC1/TD

4.23. Manual de Instalación

Cátedra	Carácter
TD	Requerido

Este documento incluye las instrucciones para realizar la instalación del producto.

4.24. Material de Apoyo al Usuario Final

Cátedra	Carácter
TD	Requerido

Corresponde a un conjunto de documentos y facilidades de uso del sistema, incluyendo: Guías del Usuario, Guías de Operación, Guías de Mantenimiento y Sistema de Ayuda en Línea.

4.25. Producto

Cada alumno deberá diseñar un software completo. Los archivos del producto empaquetados y almacenadas en un CD con los mecanismos apropiados para facilitar su instalación. El producto, a partir de la primera iteración de la fase de Construcción es desarrollado incremental e iterativamente, obteniéndose un nuevo release al final de cada iteración.

El producto se compone de los siguientes módulos:

4.25.1. Solución del negocio.

Cátedra	Carácter
TD	Requerido

Todo el software requerido para satisfacer la necesidad que originó el proyecto, acorde al tipo de negocio que le sea asignado por el docente.

4.25.2. Dígitos verificadores verticales y horizontales.

Cátedra	Carácter
TD	Requerido

La función de estos dígitos verificadores es la de verificar la integridad de los datos almacenados en la base de datos.

Es importante, al momento de determinar el algoritmo de cálculo a emplear, que este permita detectar si dos valores han sido intercambiados de posición. Por eso en el cálculo no sólo debe participar el contenido del atributo sino también la posición del carácter y la posición del atributo dentro de la entidad.

Recuerden que cuando se inicia la aplicación, y antes de dar acceso a la ventana de log-in, se debe realizar el proceso de verificación de integridad de la base de datos. En caso de error, se deberá informar al usuario para que tome las medidas adecuadas. Los dígitos verificadores horizontales se guardan en un atributo de las entidades bajo análisis mientras que los verticales se pueden guardar en una entidad adicional creada para ese fin, la cual deberá formar parte del DER de seguridad.

4.25.2.1.1. Características acerca de los dígitos verificadores que se deben incluir en la carpeta de proyecto

- Detalle de cómo se utilizarán los dígitos verificadores horizontal y vertical.
- Descripción de las operaciones de restauración a realizar en caso de error en alguno de ellos.
- Algoritmo de cálculo o código fuente a implementar para los cálculos

PROYECTO FINAL TC1/TD

4.25.3. Log-in /log-out de usuarios.

Cátedra	Carácter
TD	Requerido

Permite verificar la identidad del usuario a través del ingreso de su nombre de usuario y su clave, asignándole el perfil que le corresponda.

Se debe describir como será la política de 'log-in' y de asignación de permisos utilizando el método de Usuario, Familia y Patentes., como se hará el mantenimiento de los perfiles, tipos de patentes previstas y tipos de familias previstas.

4.25.3.1. Características del acceso al sistema que se deben incluir en la carpeta de proyecto

- Detalle preliminar de los perfiles de usuario detectados en el análisis y sus correspondientes patentes.
- Detalle preliminar de las familias a partir del análisis de los distintos perfiles.

4.25.4. Asignación de perfiles utilizando el modelo de Usuario/Familia/Patente.

Cátedra	Carácter
TD	Requerido

Este método permite la asignación rápida de los permisos y la definición de perfiles, agrupando dentro de las familias aquellos conjuntos de patentes comunes a varios usuarios.

Ante todo debemos definir que es una PATENTE y que es una FAMILIA.

Una PATENTE es un permiso otorgado a un USUARIO para acceder y ejecutar cierta parte del programa desarrollado.

Una FAMILIA es un conjunto de patentes que habilita al usuario a acceder a los permisos de cada una de las PATENTES que la forman.

Por ejemplo:

Supongamos que Crear, Modificar, Leer, Copiar, Enviar email y Borrar son nombres de PATENTES de un sistema (cuyas referencias se encuentran almacenadas en la tabla PATENTE) y que Admin y Superv son USUARIOS del sistema (con sus datos almacenados en la tabla USUARIO).

Entonces debemos definir los perfiles que asigne los permisos a cada uno de los usuarios.

En este caso asignaremos a Admin permiso para Crear, Modificar, Leer y Copiar. A Superv asignaremos Crear, Modificar y Borrar.

Usuario: Admin		Usuario: Superv
Crear,		Crear,
Modificar		Modificar
Leer		Borrar
Copiar		

La asignación de las PATENTES a cada USUARIO se almacena en la tabla de asociación USUARIO-PATENTE a través de sus respectivos códigos de identificación.

En el ejemplo anterior, las dos primeras patentes son las mismas para los dos usuarios, pero que pasaría si debemos asignar a muchos usuarios muchas patentes iguales.

PROYECTO FINAL TC1/TD

Usuario: Admin		Usuario: Superv
Crear,		Crear,
Modificar		Modificar
Leer		Borrar
Copiar		

Para simplificar esa asignación usamos las FAMILIAS. En una familia, a la que llamaremos **CAMBIAR**, agrupamos las patentes **Crear** y **Modificar**.

Todas las relaciones de pertenencia de una patente a una familia se las almacena en la tabla de asociación FAMILIA – PATENTE.

Una vez definida la familia podemos asignar a Admin la familia **CAMBIAR** y las patentes **Leer** y **Copiar**; y a Superv la familia **CAMBIAR** y la patente **Borrar**

Usuario: admin.		Usuario: Superv
CAMBIAR		CAMBIAR
Leer		Borrar
Copiar		

Las asignaciones de las familias a los usuarios quedan almacenadas en la tabla de asociación USUARIO-FAMILIA.

4.25.5. Registro de actividades & Bitácora de accesos.

Cátedra	Carácter
TD	Requerido

En ella deben quedar registradas todas las operaciones que realicen los usuarios durante la utilización del programa, lo cual permitirá hacer un trazado del recorrido realizado por un usuario dentro de la aplicación a partir de su solicitud de registro hasta que se sale del sistema.

4.25.5.1. Características de la bitácora que se deben incluir en la carpeta de proyecto

- Detalle de los datos que se almacenarán
- Detalle de las distintas formas de obtener información a partir de esos datos.

4.25.6. Multi-idioma.

Cátedra	Carácter
TD	Requerido

Debe permitir el cambio de idioma de todas las leyendas y títulos que se lean en los formularios, a partir de la selección en pantalla. El cambio debe ser dinámico, es decir que si selecciono un nuevo idioma, este debe aparecer reflejado en los formularios sin necesidad de recarga.

4.25.7. Arquitectura Escalable & Extensible. Patrones de diseño.

Cátedra	Carácter
TD	Opcional

El software deberá ser diseñado teniendo en cuenta una arquitectura en n-capas con orientación a servicios, para facilitar la portabilidad y la reutilización de código, y conseguir un bajo acoplamiento entre componentes. Asimismo, esto brinda una cierta flexibilidad en la elección del lenguaje de programación de los diferentes componentes, pudiendo elegir el que mejor solucione una problemática determinada.

PROYECTO FINAL TC1/TD

Se fomentará el uso de *patrones de diseño* para la construcción de la solución, con el fin de facilitar el diseño y lograr que la misma se base en las *buenas prácticas* del diseño y desarrollo de software.

4.25.8. Pruebas unitarias.

Cátedra	Carácter
TD	Opcional

En desarrollo del software, una prueba unitaria es una forma de probar la corrección de un módulo de código.

La idea es escribir casos de prueba para cada función no trivial o método en el módulo de forma que cada caso sea independiente del resto.

Para que una prueba unitaria sea buena se deben cumplir los siguientes requisitos:

- **Automatizable:** no debería requerirse una intervención manual. Esto es especialmente útil para *integración continua*.
- **Completas:** deben cubrir la mayor cantidad de código.
- **Repetibles:** no se deben crear pruebas que sólo puedan ser ejecutadas una sola vez. También es útil para *integración continua* y para las *pruebas de regresión*.
- **Independientes:** la ejecución de una prueba no debe afectar a la ejecución de otra.
- **Profesionales:** las pruebas deben ser consideradas igual que el código, con la misma profesionalidad, documentación, etc.

Aunque estos requisitos no tienen que ser cumplidos a rajatabla, sí que es más que recomendable el seguirlos porque sino, las pruebas pierden parte de su función.

El software a desarrollar debe estar acompañado por las pruebas unitarias suficientes para abarcar la funcionalidad de los principales módulos. Se puede optar por herramientas como [NUnit](#) para implementar esta funcionalidad.

4.25.9. Indicadores de Rendimiento del Software.

Cátedra	Carácter
TD	Opcional

Para comprobar el correcto funcionamiento del software y garantizar la escalabilidad, es necesario medir el rendimiento del software, para analizar si el mismo se está comportando de manera *saludable*. Cualquier indicador que esté fuera de los parámetros esperados puede indicar una falla en el diseño del software. Como ejemplo, podemos decir que si el software crea muchas conexiones en simultáneo a la base de datos para una única instancia de la aplicación y no las destruye, puede dar indicio de que algo en el código del componente de datos, o en los componentes que lo utilizan, no está funcionando correctamente.

La carpeta de proyecto deberá incluir datos, que pueden ser recolectados a través de algún escenario particular de las [pruebas unitarias](#), para realizar un informe de rendimiento del software, como:

- Tiempo medio de acceso a la base de datos.
- Cantidad promedio de conexiones simultáneas con la base de datos.
- Cantidad promedio de usuarios por segundo.
- Tamaño de caché de objetos.
- Porcentaje de reutilización de conexiones del pool.

4.25.10. Manejo de Excepciones.

PROYECTO FINAL TC1/TD

Cátedra	Carácter
TD	Requerido

Una excepción es causada por la ocurrencia de un error durante el tiempo de ejecución de un software. Cuando se produzca el error se puede decir que se genera una excepción.

Las razones por las que se elevará una excepción serán diversas:

- Fallos del hardware. problemas derivados del uso del hardware, como cortes en la línea de comunicación.
- Fallos producidos por el software del sistema: acceso a zonas de memoria restringidas, operaciones aritméticas incorrectas, utilización de variables no instanciadas.
- Ejecución de instrucciones que disparan a las excepciones.
- Fallos en las llamadas a funciones y procedimientos.

Por lo que una excepción se podrá elevar por causas exteriores internas a nuestro programa.

Al elevarse la excepción, nos indica que se ha producido un error y que por lógica tenemos que controlarlo para evitar problemas mayores. Para ello dispondremos de una serie de procedimientos encargados de tratar excepciones, son los llamados manejadores de excepciones.

Cuando se produzca una excepción, tomará el control el manejador asociado a ese tipo de excepción en concreto y en una zona determinada de nuestro software, es decir nos puede interesar controlar una excepción de manera distinta dependiendo de dónde nos encontremos.

PROYECTO FINAL TC1/TD

4.26. Resumen de Entregables a Desarrollar por Cátedra

Carácter	Cátedra	Entregable	Realizado	Revisado
Requerido	ASA	<ul style="list-style-type: none"> • 4.1. Descripción de la Empresa. • 4.7. Glosario. • 4.8. Modelo de Casos de Uso. • 4.9. Especificaciones de Casos de Uso. • 4.13. Diagrama de Clases. • 4.14. Diagrama de Secuencia. • 4.16. DER. 		
	TC1	<ul style="list-style-type: none"> • 4.6. Estudio de viabilidad. <ul style="list-style-type: none"> ◦ 4.6.1. Legal. ◦ 4.6.2. Técnica. ◦ 4.6.3. Operativa. ◦ 4.6.4. Económico/Financiera. • 4.11. Mapa de Navegación. • 4.12. Prototipo de Interfaces de Usuario. • 4.15. Diagrama de Componentes. 		
	TD	<ul style="list-style-type: none"> • 4.23. Manual de Instalación • 4.24. Material de apoyo al Usuario Final • 4.25. Producto <ul style="list-style-type: none"> ◦ 4.25.1. Solución del negocio. ◦ 4.25.2. Dígitos verificadores verticales y horizontales. ◦ 4.25.3. Log-in/Log-out de usuarios. ◦ 4.25.4. Asignación de perfiles utilizando el modelo de Usuario/Familia/Patente. ◦ 4.25.5. Registración de actividades & bitácora de accesos. ◦ 4.25.6. Mutli-idioma. ◦ 4.25.10. Manejo de excepciones. 		
Opcional	ASA	<ul style="list-style-type: none"> • 4.2. Plan de Desarrollo del Software. • 4.3. Visión. • 4.4. Modelo de Casos de Uso del Negocio. • 4.5. Modelo de Dominio/Objetos del Negocio. • 4.10. Especificaciones Adicionales. • 4.19. Solicitud de Cambio. • 4.20. Plan de Iteración. • 4.21. Evaluación de Iteración. 		
	TC1	<ul style="list-style-type: none"> • 4.6. Estudio de viabilidad. <ul style="list-style-type: none"> ◦ 4.6.5. De Gestión. ◦ 4.6.6. Comercial. • 4.10. Especificaciones Adicionales. • 4.17. Modelo de Despliegue. • 4.18. Especificación de Casos de Prueba. • 4.19. Solicitud de Cambio. • 4.22. Lista de Riesgos. 		
	TD	<ul style="list-style-type: none"> • 4.25. Producto <ul style="list-style-type: none"> ◦ 4.25.7. Arquitectura escalable & extensible. Patrones de Diseño. ◦ 4.25.8. Pruebas unitarias. ◦ 4.25.9. Indicadores de rendimiento del software. 		

4.27. Matriz de Trazabilidad entre Entregables y Proyecto Modelo

Entregable	Proyecto Modelo																		
	Gestión del Proyecto					Modelado del Negocio			Requisitos					A&D	Implementación				
	5.2.1	5.2.2	5.2.3	5.2.4	5.3.1	5.3.2.1	5.3.2.2	5.3.2.3	5.4.1	5.4.2	5.4.3	5.4.4	5.4.5	5.4.6	5.4.7	5.5	5.6.1	5.6.2	5.6.3
4.1. Descripción de la Empresa					R														
4.2. Plan de Desarrollo del Software	R																		
4.3. Visión												O							
4.4. Modelo de Casos de Uso del Negocio						O													
4.5. Modelo de Dominio / Objetos del Negocio							O	O											
4.6. Estudio de Viabilidad																			
4.6.1. Vialidad Legal					R														
4.6.2. Vialidad Técnica					R														
4.6.3. Vialidad Operativa					R														
4.6.4. Vialidad Económico/Financiera					R														
4.6.5. Vialidad de Gestión.					O														
4.6.6. Vialidad Comercial					O														
4.7. Glosario												R							
4.8. Modelo de Casos de Uso														R					
4.9. Especificaciones de Casos de Uso.														R					
4.10. Especificaciones Adicionales														O					
4.11. Mapa de Navegación																R			
4.12. Prototipos de Interfaces de Usuario																R			
4.13. Diagrama de Clases																R			
4.14. Diagrama de Secuencia																R			
4.15. Diagrama de Componentes																	R		
4.16. DER																R			
4.17. Modelo de Despliegue																	O		
4.18. Especificación de Casos de Prueba																O			
4.19. Solicitud de Cambio														O					
4.20. Plan de Iteración						O													
4.21. Evaluación de Iteración						O													
4.22. Lista de Riesgos			O																R
4.23. Manual de Instalación																			R
4.24. Material de Apoyo al Usuario Final																			R
4.25. Producto																			
4.25.1. Solución del negocio.																			R
4.25.2. Dígitos verificadores verticales y horizontales.																			R
4.25.3. Log-in /log-out de usuarios.																			R
4.25.4. Usuario/Familia/Patente.																			R
4.25.5. Registro de actividades & Bitácora de accesos.																			R
4.25.6. Multi-idioma.																			R
4.25.7. Arq. Escalable & Extensible. Patrones Diseño.																O			
4.25.8. Pruebas unitarias.																O			
4.25.9. Indicadores de Rendimiento del Software.																O			
4.25.10. Manejo de Excepciones.																	O		R

5. Proyecto Modelo

El objetivo de esta sección es mostrar un ejemplo de desarrollo de software basado en la metodología de [Proceso Unificado](#). El proyecto es el desarrollo de un sistema para la gestión de artículos deportivos de una empresa del sector de ventas de deportes a clientes tanto a mayoristas como a minoristas.

Se incluye hasta la segunda iteración de la fase de construcción, según la división establecida en el documento Plan de Desarrollo Software. Por motivos de privacidad no se pueden publicar los datos de la entidad para la que se diseñó el software.

5.1. Índice

La carpeta deberá contar con un índice completo donde se encuentren detallados cada uno de los títulos y subtítulos de la carpeta, acompañados por la página en la que se encuentran.

El índice es la guía que permite ubicar los contenidos de un trabajo. Por lo tanto debe ser coherente con el contenido de la carpeta.

5.2. Gestión del Proyecto

En el apartado de **Gestión del Proyecto** se muestran las planificaciones temporales de desarrollo del proyecto en su fase de inicio y de elaboración, así como el diario de ejecución del proyecto, junto con el diario de construcción de la aplicación y cumplimiento de los plazos estimados.

5.2.1. [Plan de Desarrollo de Software](#)

5.2.2. Estudio de Viabilidad

5.2.2.1. Viabilidad Legal

La empresa deberá poseer las licencias correspondientes al sistema operativo Windows XP, motor de base de datos y los componentes adicionales que se requieran para su funcionamiento, incluyendo generador de reportes de usuarios finales, visualizadores de archivos en formato PDF y otros en cada una de las computadoras cliente o servidores, de acuerdo al caso, en que se instale el software.

El sistema generará todos los reportes y archivos exportables necesarios para las auditorias realizadas por la Administración Federal de Ingresos Públicos (A.F.I.P) y no permite alteración alguna sobre la registración de comprobantes fiscales ya registrados en el software, o la duplicidad de numeración de cada uno de ellas.

El software brindará la certeza de que el mismo realizará un correcto uso y protección de la información almacenada, ya que constituye uno de sus principales activos intangibles.

El software implementa las medidas necesarias para cumplir con la ley de Protección de los Datos Personales (25.326) que establece en su artículo 9º, la obligación de las empresas de proteger la información referida a personas físicas o de existencia ideal contenida en sus archivos y registros, y ordena al responsable o usuario de los mismos a adoptar las medidas técnicas y organizativas necesarias para garantizar la seguridad y confiabilidad de los datos. A su vez, prohíbe registrar datos en archivos, registros o bancos que no reúnan condiciones técnicas de integridad y seguridad.

Para cumplir con estos requisitos se adopta la norma ISO 17.799, Standard de seguridad internacionalmente reconocido, que contiene una serie de controles que contemplan las "mejores prácticas" en seguridad de la información entre las cuales se encuentra el uso de software.

De acuerdo a lo expuesto anteriormente, no se observan problemas legales en la implementación del mismo.

5.2.2.2. Viabilidad Técnica

5.2.2.2.1. Situación actual

El equipamiento técnico con el cual cuenta el sector afectado al cambio del sistema, consta de:

Cantidad	Descripción
12	PC con Microprocesador INTEL Pentium 4 de 2.66 Gb RAM 512 Mb. a 400 Mhz Placa de sonido multimedia Placa de red PCI Wireless LINKSYS G Monitor LCD 17"
3	Impresoras Samsung Laser ML-1610
3	Impresora Epson CX 4700 Multifunción
5	Modems con Voice 56 Kb. V90 Formato PCI

PROYECTO FINAL TC1/TD

10	líneas telefónicas
----	--------------------

5.2.2.2.2. Requerimientos Técnicos

Teniendo en cuenta lo relevado, no se requiere el agregado de computadoras personales en las oficinas o depósitos de la empresa.

5.2.2.2.3. Sistema de alimentación

Para mayor seguridad ante fallos en la red de suministro eléctrico, es recomendable la instalación de un sistema de provisión de energía eléctrica con baterías, con autonomía de hasta 30 minutos.

Cantidad	Descripción
18	Equipo UPS para las PC e impresoras nuevas (en oficinas)

O bien:

Cantidad	Descripción
1	Sistema UPS general para alimentar a todas las PC de las oficinas.

NOTA: La autonomía requerida es de 1 hora ya que es tiempo suficiente como para poner en servicio el grupo electrógeno con que cuenta la emrpesa

5.2.2.2.4. Instalación

En caso de aceptar la propuesta, se ofrece lo siguiente:

- Configuración de equipos: Provisión de la mano de obra especializada para la puesta en servicio de la aplicación ofrecida, configurando los equipos y software necesario.
- Capacitación: Se capacitará al personal que la empresa designe en la operación de los equipos provistos, durante el período de 6hs. durante 6 días, luego de la puesta en funcionamiento de los equipos, a convenir en el horario de común acuerdo.

Por lo tanto, se concluye que no existen impedimentos técnicos para llevar a cabo el proyecto.

5.2.2.3. Viabilidad Operativa

Es requisito imprescindible que todo usuario que interactúe con el sistema posea conocimientos básicos de manejo de PC bajo el entorno Windows XP en el caso de usuarios finales, y conocimientos de Windows 2003 Server y SQL Server para el caso de los administradores del sistema o encargados de operaciones.

Además debe tener los conocimientos funcionales básicos del circuito administrativo de seguros para operar de manera eficiente el software.

5.2.2.4. Viabilidad Económico/Financiera

N/D.

5.2.2.5. Viabilidad de Gestión

El grupo de desarrollo del proyecto para cumplimentar con los requisitos establecidos en el apartado de Relevamiento, debe considerar la adquisición de personal capacitado para el desarrollo del software.

El grupo posee buen dominio de las herramientas de análisis, diseño y desarrollo elegidas para la realización del proyecto, y posee la experiencia en tareas similares y las capacidades necesarias por su desempeño anterior en empresas de gran magnitud y en proyectos de software de dimensiones similares.

Se deja en claro entonces que la capacidad de gestión del grupo de desarrollo es la necesaria para abordar este proyecto y completar el mismo de forma exitosa, ya que cuenta con experiencia suficiente en el desarrollo e implementación de sistemas de información y cuenta además con la tecnología y recursos requeridos para cumplir con los objetivos planteados.

5.2.2.6. Viabilidad Comercial

Al ser un producto a desarrollar para un cliente en particular, este punto no tiene validez para el presente proyecto.

5.2.3. Lista de Riesgos

N/D.

5.2.4. Planificación del Proyecto

5.2.4.1. Fase de Inicio

Plan de Iteración. Iteración de Inicio #I1

Evaluación de Iteración de Inicio #I1

5.2.4.2. Fase de Elaboración

Plan de Iteración. Iteración de Elaboración #E1

Evaluación de Iteración de Elaboración #E1

Plan de Iteración. Iteración de Elaboración #E2

Evaluación de Iteración de Elaboración #E2

Plan de Iteración. Iteración de Elaboración #E3

Evaluación de Iteración de Elaboración #E3

5.2.4.3. Fase de Construcción

Plan de Iteración. Iteración de Construcción #C1

Evaluación de Iteración de Construcción #C1

Plan de Iteración. Iteración de Construcción #C2

Evaluación de Iteración de Construcción #C2

Plan de Iteración. Iteración de Construcción #Cn

Evaluación de Iteración de Construcción #Cn

5.2.4.4. Fase de Transición

Plan de Iteración. Iteración de Transición #T1

Evaluación de Iteración de Transición #T1

Plan de Iteración. Iteración de Transición #T2

Evaluación de Iteración de Transición #T2

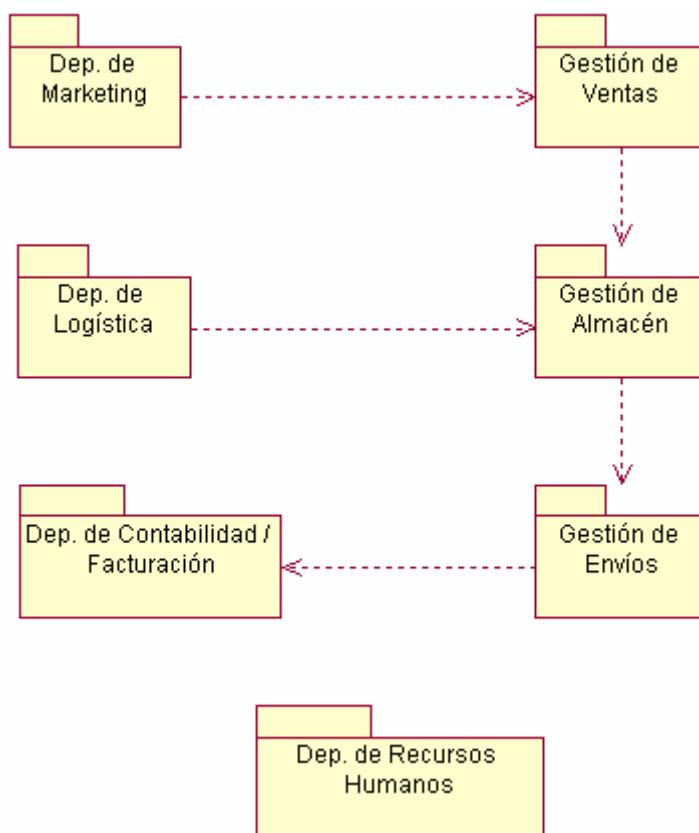
5.3. Modelado del negocio

En el apartado de **Modelado del Negocio** se encuentran los artefactos utilizados de la metodología de Proceso Unificado para definir un modelo del negocio, modelos de objetos del negocio y el modelo del dominio.

5.3.1. Descripción de la empresa

La empresa de deportes que solicitó el proyecto de desarrollo software consta de varios departamentos centralizados, un almacén central y de diversas sucursales de ventas repartidas en distintos países. Cada sucursal de ventas dispone de un almacén regional que suministra los pedidos de los clientes a los países que conforman una región determinada, siendo el almacén central el que abastece al resto de almacenes.

El diagrama que representa los diferentes subsistemas en los que se ha dividido la empresa a nivel de abstracción es el siguiente:

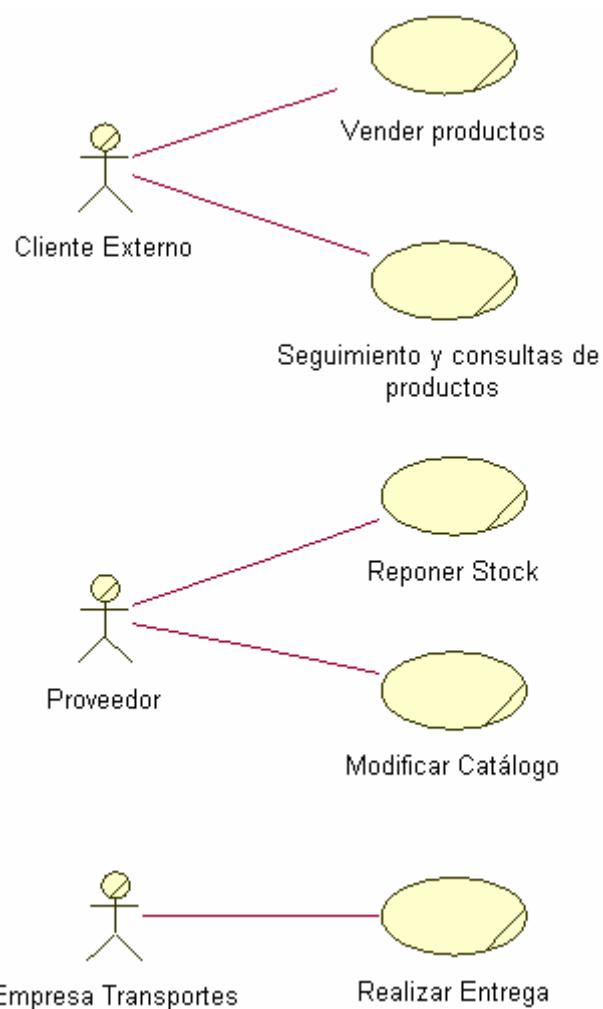


5.3.2. Modelado del Negocio

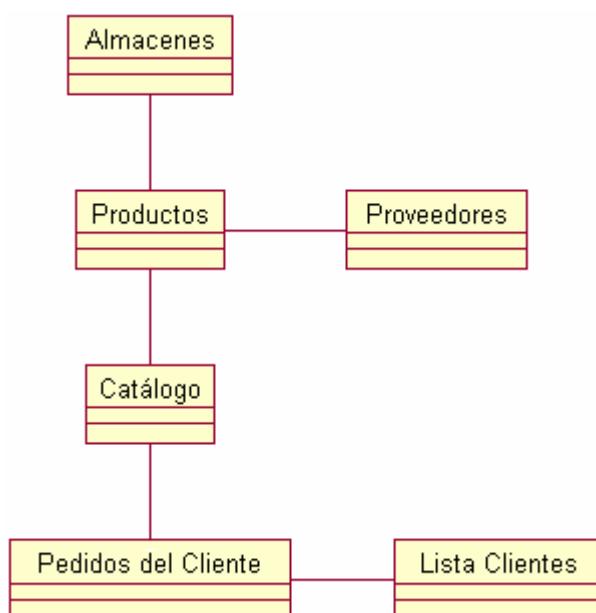
El modelado del negocio se basa en tres diagramas principales, el **modelo de casos de uso del negocio**, el **modelo del dominio** y los **modelos de objetos del negocio**.

La empresa interactúa con distintos elementos externos, entre los que se identifican el cliente externo (persona o entidad que solicita la compra de productos a la empresa), el proveedor (persona o entidad que reabastece de productos a la empresa) y por último la empresa de transportes, que es una empresa subcontratada encargada de servir los pedidos desde los distintos almacenes regionales a los clientes de la empresa.

5.3.2.1. Modelo de Casos de Uso del Negocio



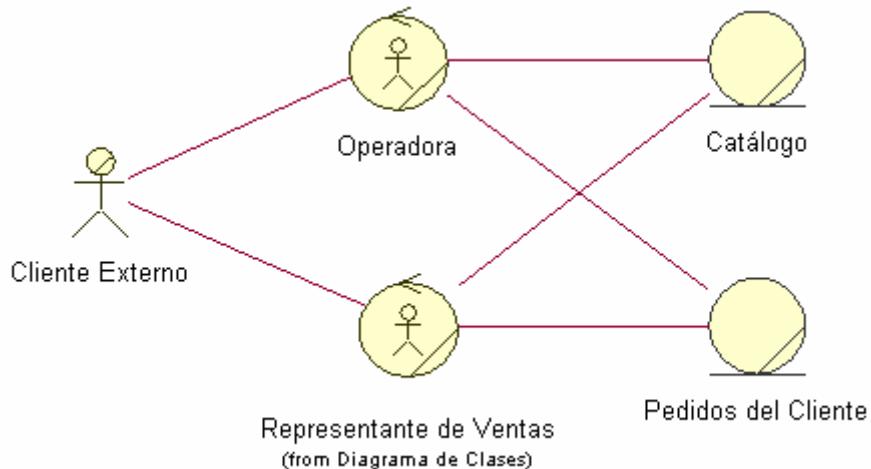
5.3.2.2. Modelo del Dominio



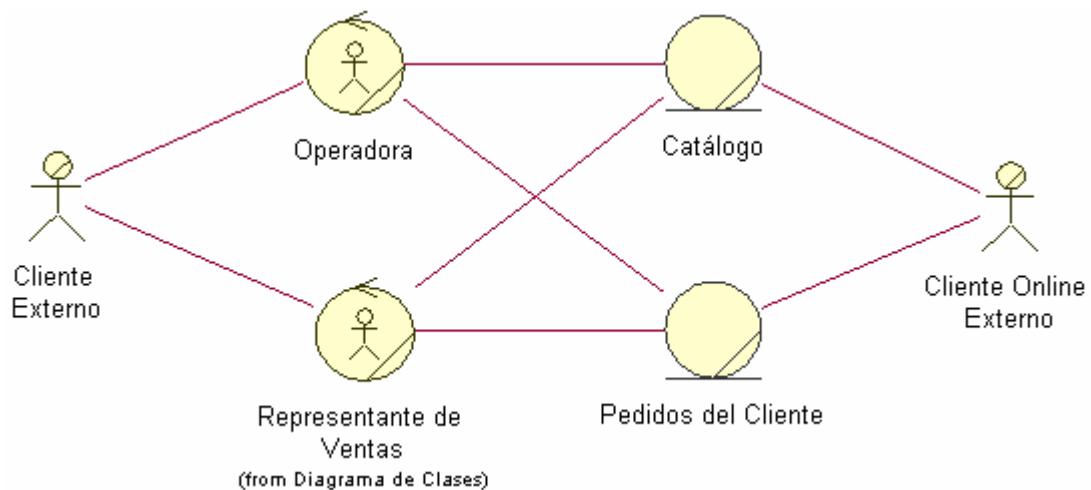
5.3.2.3. Modelo de Objetos del Negocio

Los modelos de objetos del dominio están asociados a cada uno de los casos de uso del negocio. Por ser de mayor prioridad para la empresa, el caso de uso para el cual se desarrolló el modelo de objetos fue el del caso de uso del negocio "vender productos".

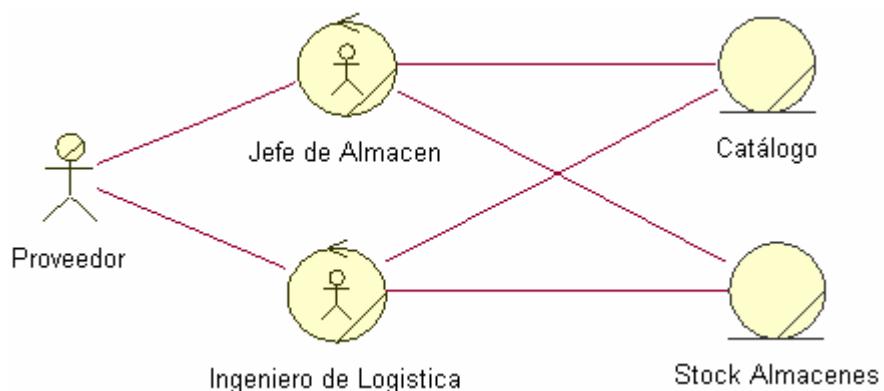
5.3.2.3.1. Modelo de Objetos de Vender Productos



5.3.2.3.2. Modelo de Objetos de Seguimiento y Consulta de Productos



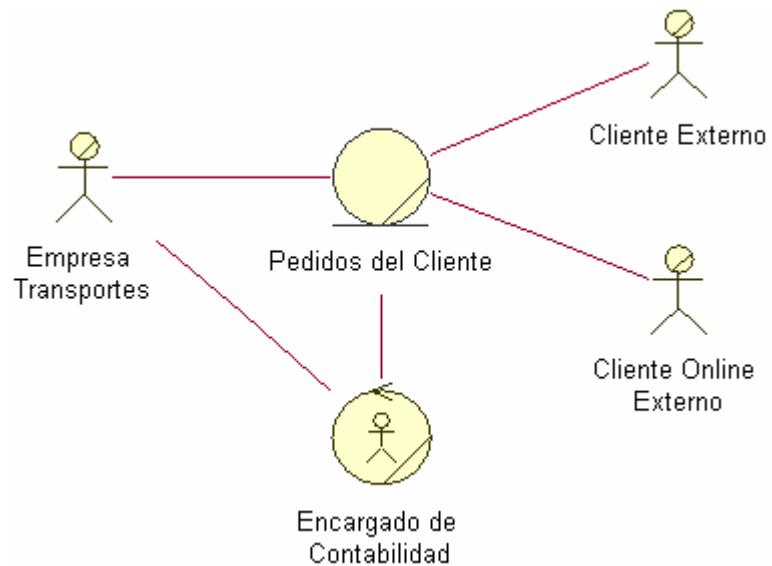
5.3.2.3.3. Modelo de Objetos de Reponer Stock



5.3.2.3.4. Modelo de Objetos de Modificar Catálogo



5.3.2.3.5. Modelo de Objetos de Realizar Entrega



5.4. Requisitos

En el apartado **Requisitos** consta de los artefactos definidos según la metodología de Proceso Unificado, es decir, el documento **Plan de Desarrollo Software**, el documento **Visión**, el documento **Glosario** y las especificaciones tanto de los casos de uso como de los casos de pruebas relacionados con estos. También se recoge la gestión del proyecto con la herramienta de Rational, el Requisite Pro, con la que además de llevar el control de toda la documentación, se puede seguir la trazabilidad entre requerimientos de todo el proyecto. En este apartado se muestran las matrices de atributos de todos los requerimientos así como la navegabilidad entre ellos. Por añadidura también se muestran los casos de uso de cada subsistema generados con la herramienta Rational Rose, y desde los cuales también se puede consultar la especificación del caso de uso.

5.4.1. [Documento Visión](#)

5.4.2. [Documento Glosario](#)

5.4.3. Solicitud de Cambio

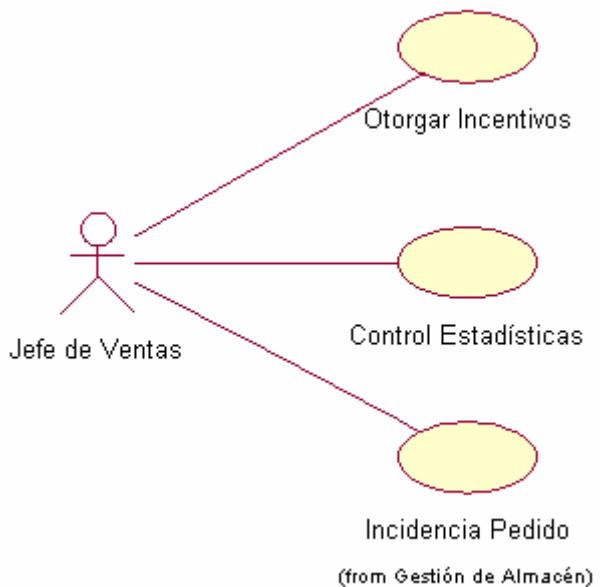
N/D.

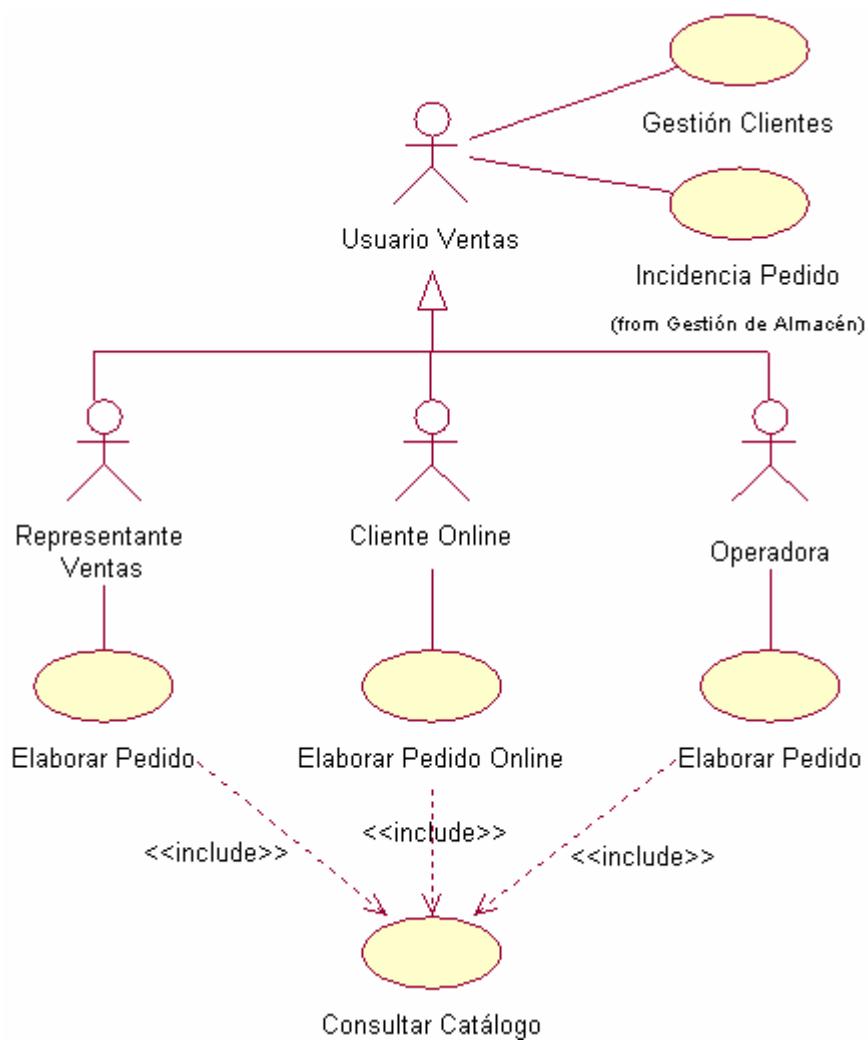
5.4.4. Modelo de Casos de Uso

A continuación se presentan los diagramas de casos de uso planteados para cada uno de los subsistemas definidos para la empresa. Se puede consultar la especificación de cada caso de uso haciendo clic sobre el diagrama del caso de uso correspondiente. Cabe destacar que los casos de uso que no se incluyeron en la fase de construcción sólo figuran en estado de propuestos, por tanto en sus primeras versiones.

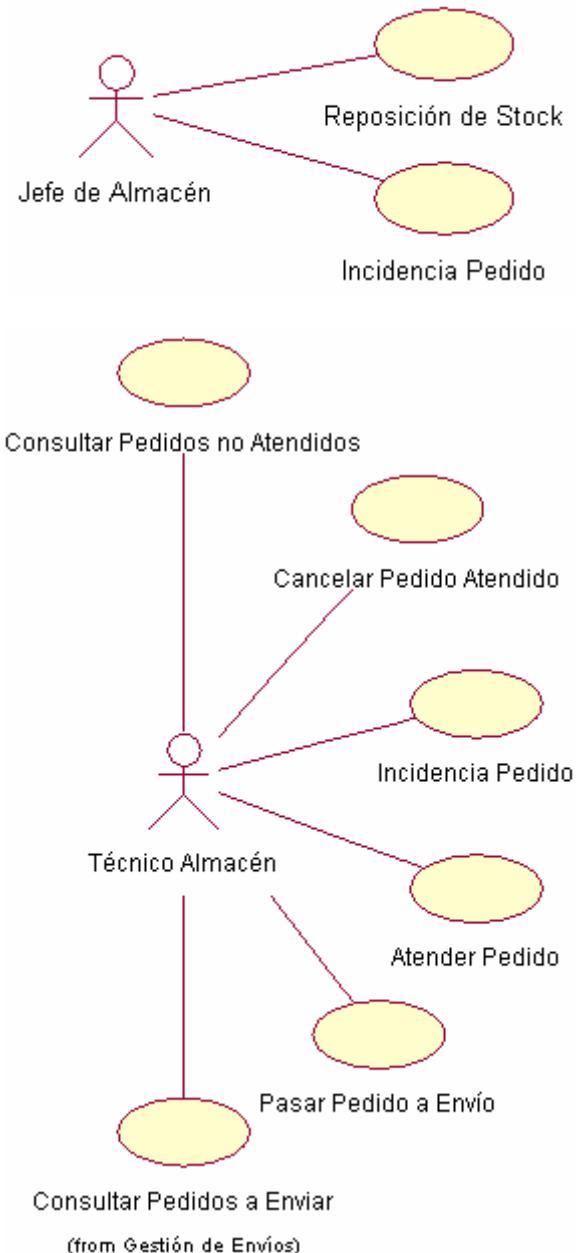
5.4.4.1. Gestión de Ventas

En el subsistema gestión de ventas participan tres actores para los cuales se generan distintos casos de uso, que se muestran a continuación.

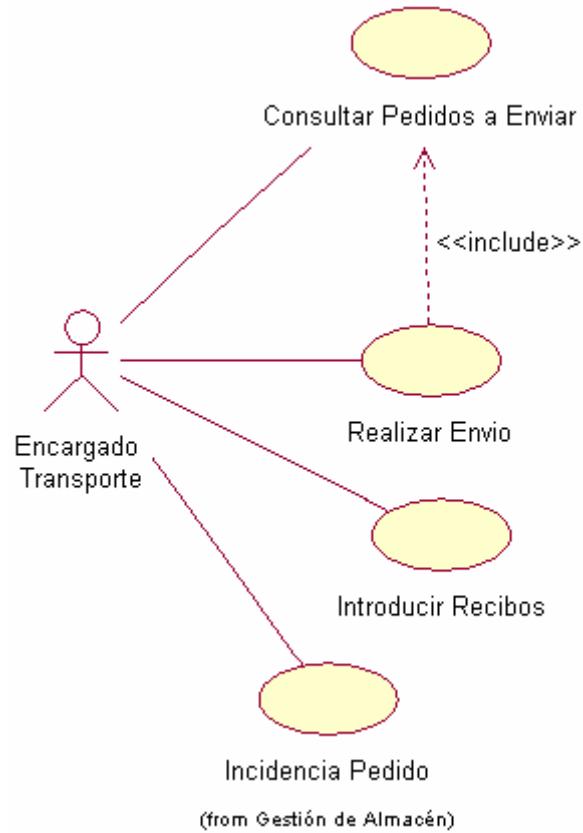




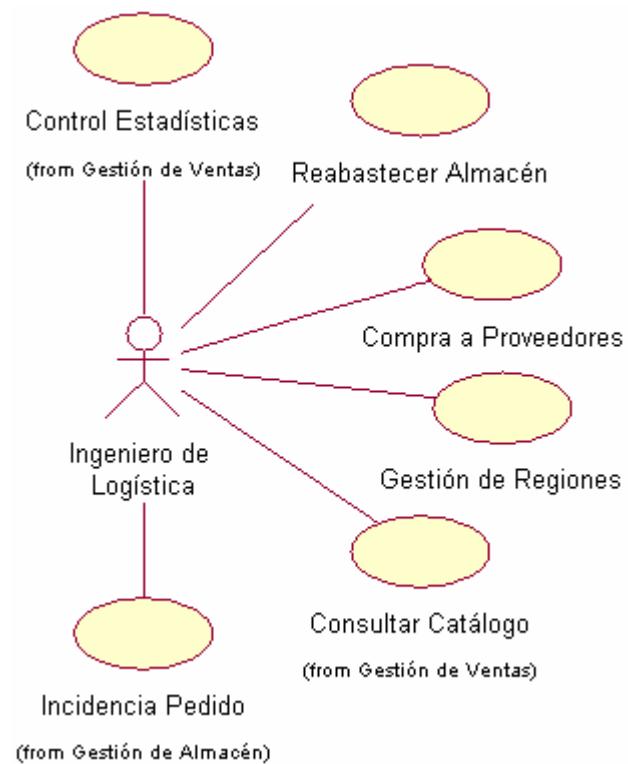
5.4.4.2. Gestión de Almacén



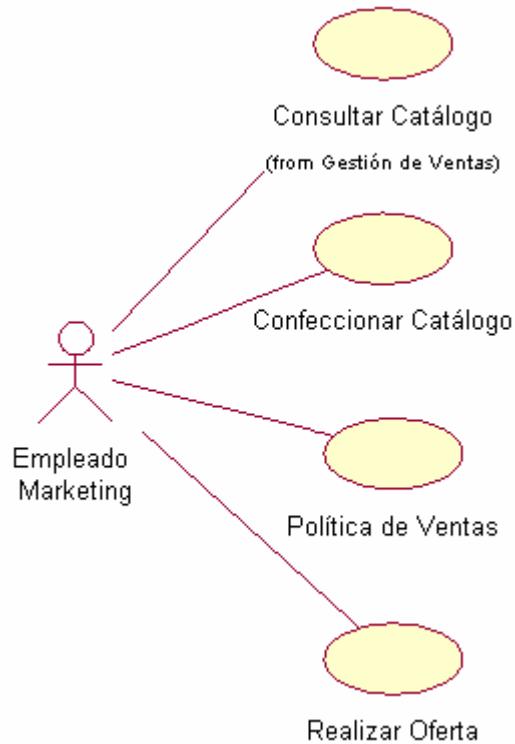
5.4.4.3. Gestión de Envíos



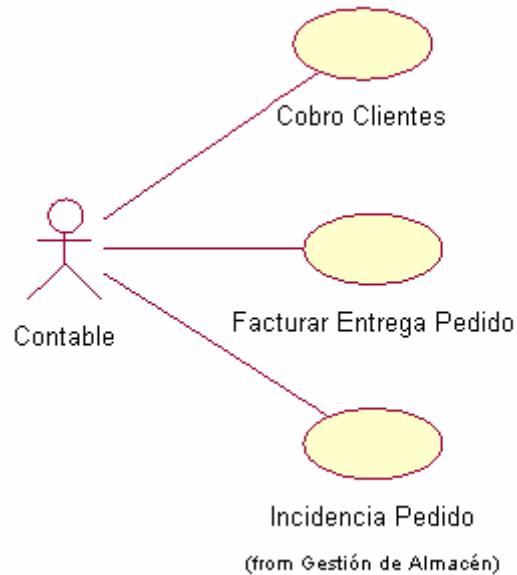
5.4.4.4. Departamento de Logística



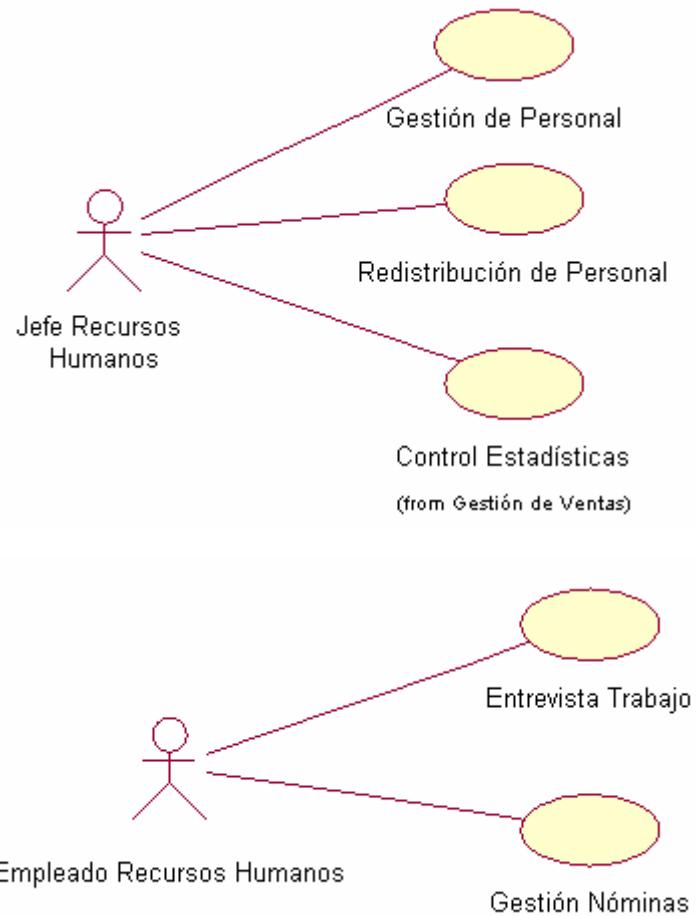
5.4.4.5. Departamento de Marketing



5.4.4.6. Departamento de Contabilidad/Facturación



5.4.4.7. Departamento de Recursos Humanos



5.4.5. Especificación de Casos de Uso

5.4.5.1. Gestión de Ventas

- 5.4.5.1.1. [Especificación del caso de uso "control de estadísticas"](#)
- 5.4.5.1.2. [Especificación del caso de uso "consultar catálogo"](#)
- 5.4.5.1.3. [Especificación del caso de uso "otorgar incentivos"](#)
- 5.4.5.1.4. [Especificación del caso de uso "elaborar pedido"](#)
- 5.4.5.1.5. [Especificación del caso de uso "elaborar pedido online"](#)
- 5.4.5.1.6. [Especificación del caso de uso "gestión de clientes"](#)

5.4.5.2. Gestión de Almacén

- 5.4.5.2.1. [Especificación del caso de uso "consultar pedidos no atendidos"](#)
- 5.4.5.2.2. [Especificación del caso de uso "atender pedido"](#)
- 5.4.5.2.3. [Especificación del caso de uso "cancelar pedido atendido"](#)
- 5.4.5.2.4. [Especificación del caso de uso "incidencia pedido"](#)
- 5.4.5.2.5. [Especificación del caso de uso "pasar pedido a envío"](#)
- 5.4.5.2.6. [Especificación del caso de uso "reposición de stock"](#)

5.4.5.3. Gestión de Envíos

- 5.4.5.3.1. [Especificación del caso de uso "consultar pedidos a enviar"](#)
- 5.4.5.3.2. [Especificación del caso de uso "introducir recibos"](#)
- 5.4.5.3.3. [Especificación del caso de uso "realizar envío"](#)

5.4.5.4. Departamento de Logística

- 5.4.5.4.1. [Especificación del caso de uso "compra a proveedores"](#)
- 5.4.5.4.2. [Especificación del caso de uso "gestión de regiones"](#)
- 5.4.5.4.3. [Especificación del caso de uso "reabastecer almacén"](#)

5.4.5.5. Departamento de Marketing

- 5.4.5.5.1. [Especificación del caso de uso "confeccionar catálogo"](#)
- 5.4.5.5.2. [Especificación del caso de uso "política de ventas"](#)
- 5.4.5.5.3. [Especificación del caso de uso "realizar oferta"](#)

5.4.5.6. Departamento de Contabilidad/Facturación

- 5.4.5.6.1. [Especificación del caso de uso "cobro a clientes"](#)

5.4.5.7. Departamento de Recursos Humanos

- 5.4.5.7.1. [Especificación del caso de uso "entrevista de trabajo"](#)
- 5.4.5.7.2. [Especificación del caso de uso "gestión de nóminas"](#)
- 5.4.5.7.3. [Especificación del caso de uso "gestión de personal"](#)
- 5.4.5.7.4. [Especificación del caso de uso "redistribución de personal"](#)

5.4.6. Especificaciones Adicionales

N/D.

5.4.7. Especificación de Casos de Prueba

En el apartado **Especificación de Casos de Prueba** se encuentran las especificaciones de casos de pruebas funcionales. Se muestran únicamente los casos de pruebas generados para los casos de uso incorporados hasta la segunda iteración de la fase de construcción.

5.4.7.1. Base de datos

5.4.7.1.1. [Especificación de la Base de Datos de Prueba](#)

5.4.7.2. Gestión de almacén

5.4.7.2.1. [Especificación de Caso de Prueba "Consultar pedidos no atendidos"](#)

5.4.7.2.2. [Especificación de Caso de Prueba "Atender Pedido"](#)

5.4.7.2.3. [Especificación de Caso de Prueba "Cancelar Pedido Atendido"](#)

5.4.7.2.4. [Especificación de Caso de Prueba "Incidencia Pedido"](#)

5.4.7.2.5. [Especificación de Caso de Prueba "Pasar Pedido a Envío"](#)

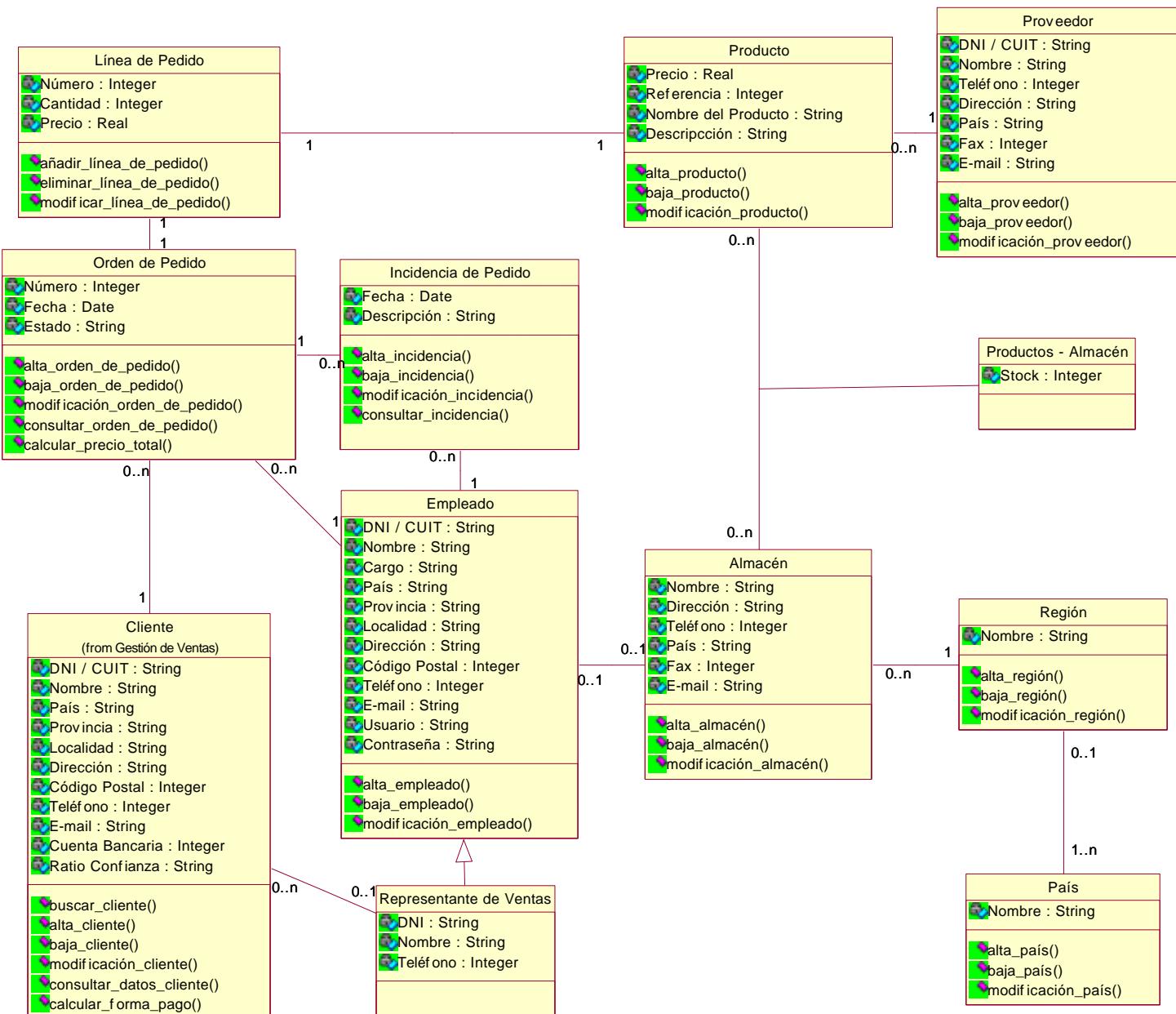
5.4.7.3. Gestión de Ventas

5.4.7.3.1. [Especificación de Caso de Prueba "Elaborar Pedido"](#)

5.5. Análisis/Diseño

En el apartado **Análisis/Diseño** se muestran tanto el modelo de análisis/diseño (diagrama de clases, diagrama de secuencia) como el modelo de datos (modelo entidad - relación), desde los cuales se puede consultar la especificación de los métodos de clase más relevantes o las especificaciones de atributos.

5.5.1. Diagrama de Clases



5.5.2. Diagramas de Secuencia

N/D.

5.5.3. Diagramas de Transición-Estados

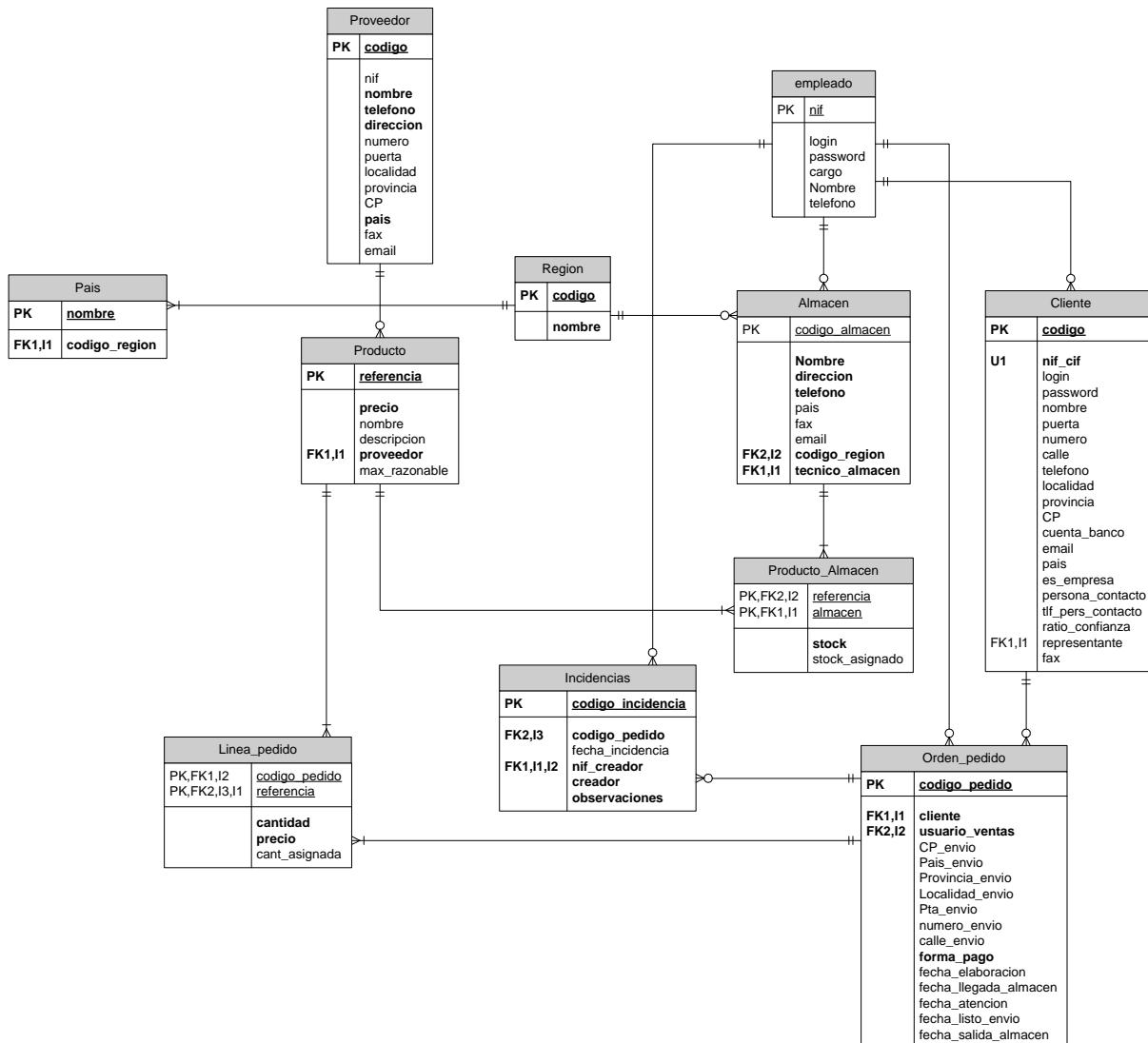
N/D.

5.5.4. Diagramas de Actividad

N/D

PROYECTO FINAL TC1/TD

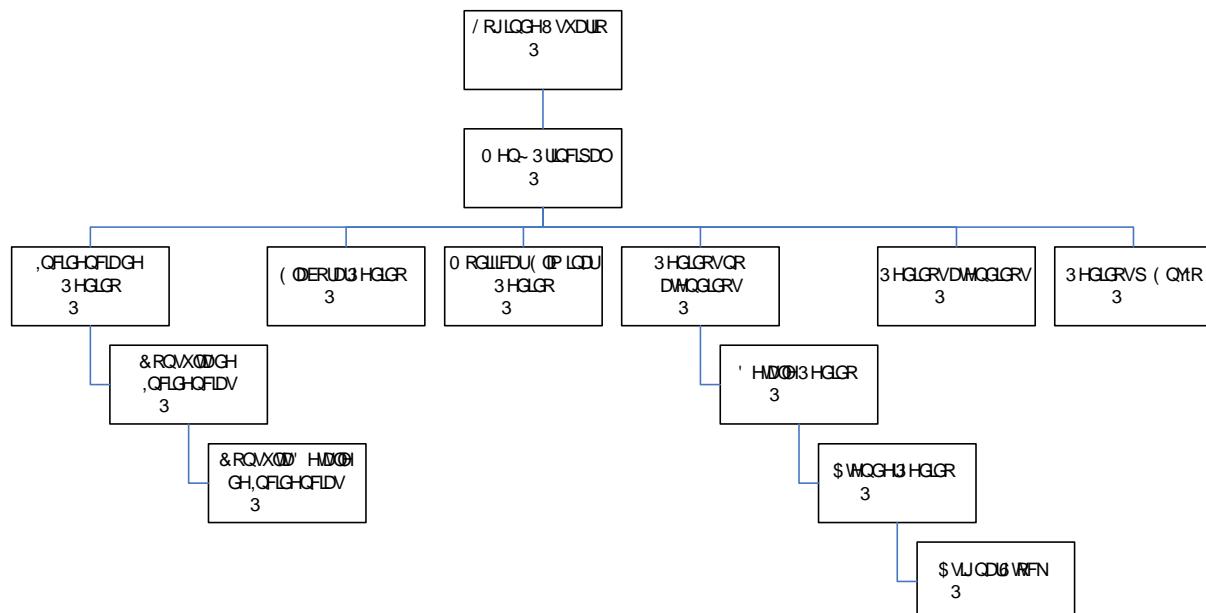
5.5.5. DER



5.6. Implementación

En el apartado **Implementación** se muestran los prototipos de interfaces de usuario de la aplicación, tanto para el sistema de gestión de ventas como para el sistema de gestión de almacén. También en este apartado se muestran los diagramas de componentes y diagrama de despliegue que modela las aplicaciones incorporadas en el proyecto hasta la segunda iteración de la fase de construcción (según la definición de fases e iteraciones de la metodología RUP) y desde los cuales, a través de los componentes se puede consultar el código fuente de cada uno.

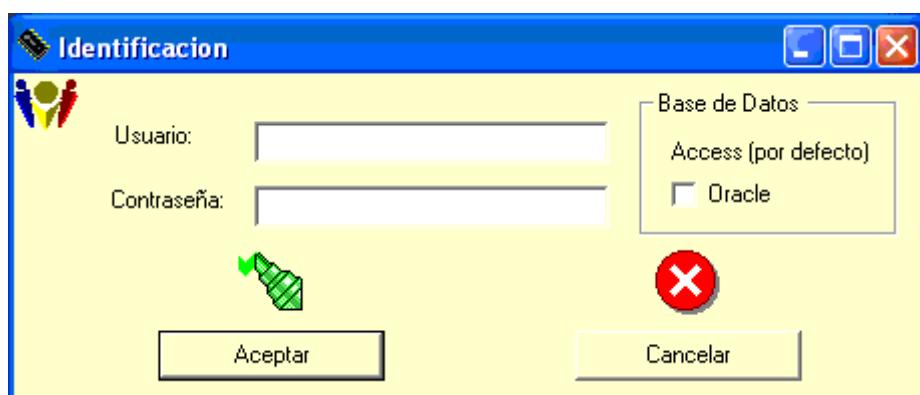
5.6.1. Mapa de Navegación



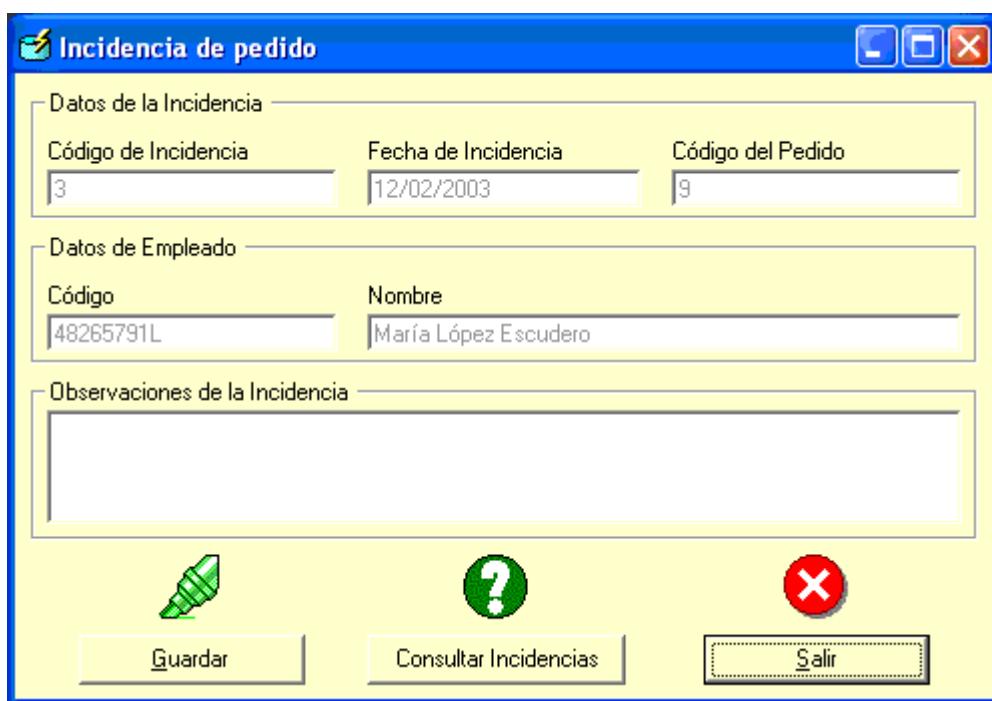
5.6.2. Prototipo de Interfaces de Usuario

5.6.2.1. Interfaces Comunes

La aplicación de cualquier subsistema de la empresa dispone de una primera ventana de identificación del usuario. Sólo usuarios registrados en la base de datos pueden acceder al sistema. La interfaz que se presenta en la identificación es la siguiente:



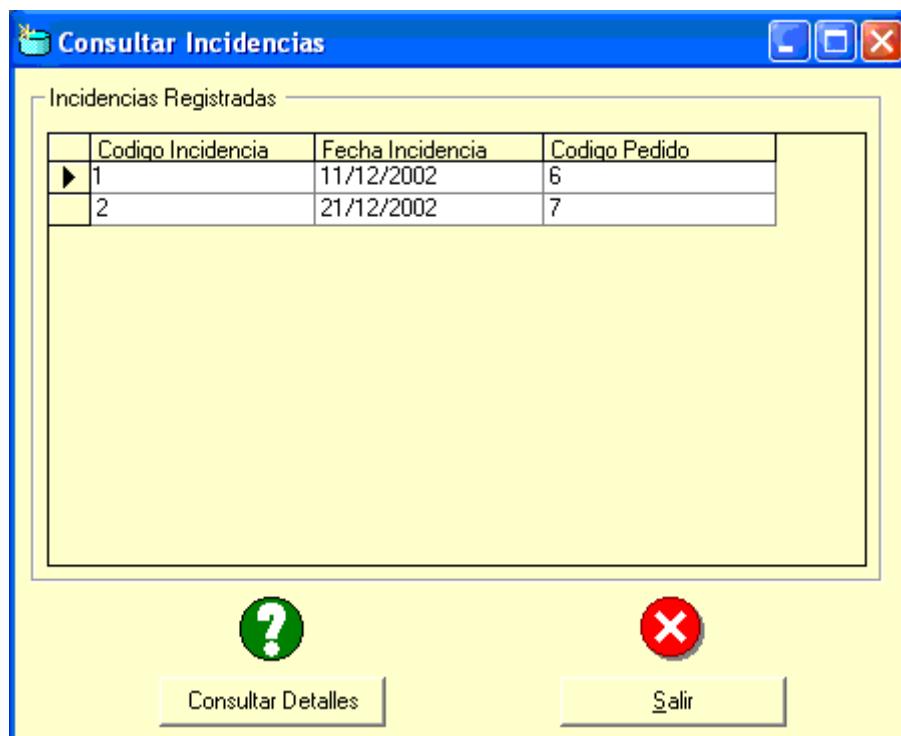
En caso de seleccionar incidencia pedido, la interfaz gráfica que se mostrará será la siguiente:



En la consulta de incidencias se podrá ver una lista de las incidencias registradas en el sistema.

En las siguientes iteraciones de construcción se implementarán casos de uso como el de gestión de clientes (botón que aparece en la pantalla de los datos del cliente pero que no tiene ninguna funcionalidad asociada).

La interfaz es el siguiente:



Para consultar los detalles de una incidencia el prototipo de interfaz gráfica es el siguiente:

Detalles de Incidencia

Datos de la Incidencia		
Código de Incidencia	Fecha de Incidencia	Código del Pedido
1	11/12/2002	6
Datos de Empleado		
Código	Nombre	
26485963M	Antonio Milla López	
Observaciones de la Incidencia		
Reponer stock, producto 1 por debajo del minimo		

 [Salir](#)

5.6.2.2. Gestión de Ventas

Para el subsistema de ventas el prototipo de interfaz gráfica es el de elaborar pedido:

PROYECTO FINAL TC1/TD

Elaborar Pedido

Datos del Representante de Ventas

Código	Nombre	
48265791L	María López Escudero	Gestión de Clientes

Datos cliente

Código cliente	Dni/Cif		Persona de Contacto		
Nombre		Buscar	Teléfono contacto		
Calle	Nº	Pta	Localidad	Provincia	CP
Teléfono	Fax	E-mail	País		

Pedidos En Elaboración:

Código	Fecha Elaboración

[Nuevo](#) [Modificar](#) [Cancelar Pedido](#)

Pedidos Enviados a Almacén

Código	Fecha elaboración	Fecha llegada almacén

[Consultar Pedido](#) [Salir](#)

El usuario de ventas puede generar un pedido nuevo para un cliente, modificar un pedido que esté en elaboración, consultar pedidos en elaboración, cancelar pedidos o consultar el detalle de pedidos ya enviados al almacén.

En el caso de elaborar un pedido nuevo o de modificar uno existente la interfaz gráfica que se presentará será la siguiente:

Elaborar Pedido Nuevo

Datos del Pedido		Codigo Representante/Operadora	
Código pedido	Fecha	48265791L	
9	12/02/2003		
Dirección Envío	Nº	Pta	
C/ Melias	3	25	
Provincia	Localidad	CP	País
Valencia	Manises	46985	España

Productos

Nueva línea			
Referencia	Nombre	Cantidad	Precio
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Referencia	Nombre	Cantidad	Precio	Total
<input type="text"/>				

Forma de pago

IVA	<input type="text"/>
Total	<input type="text"/>

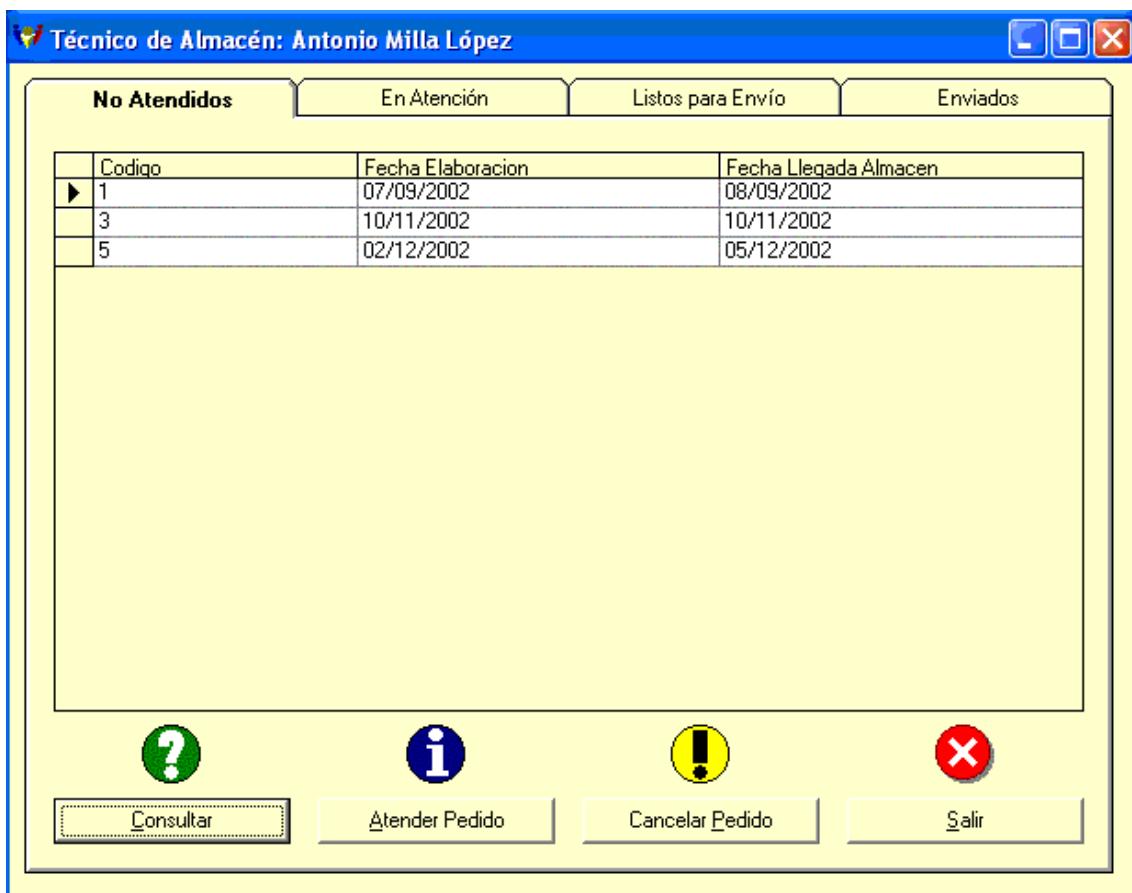
   

5.6.2.3. Gestión de Almacén

Para la gestión de almacén el prototipo de interfaz gráfica es el siguiente, en el que se pueden observar cuatro pestañas principales, una para no atendidos (pedidos en estado de no atención), otra para en atención (pedidos para los cuales ha sido reservado stock)

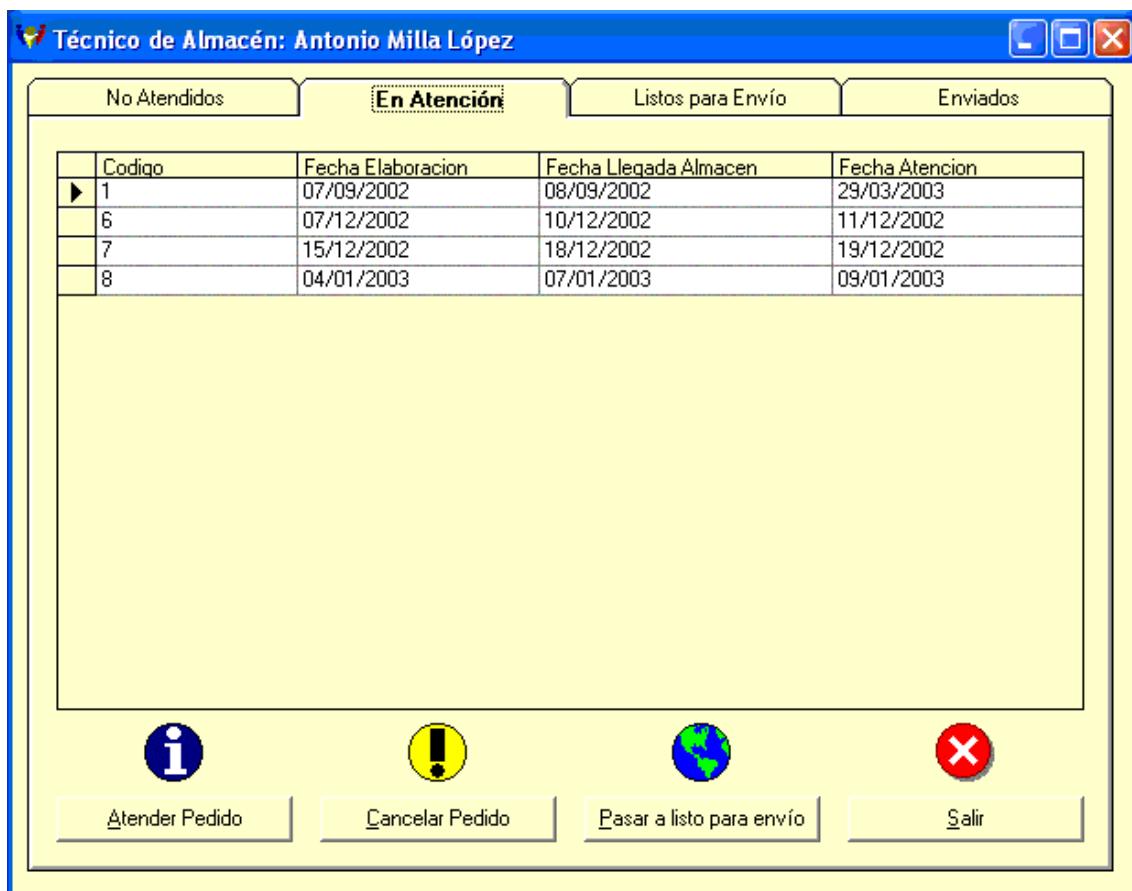
En la pestaña de no atendidos el técnico de almacén puede realizar las operaciones de consulta de detalles de un pedido, puede atender directamente el pedido seleccionado, puede cancelar el pedido seleccionado o salir de nuevo a la interfaz de identificación.

La interfaz gráfica principal de técnico de almacén en no atendidos es la siguiente:



En la pestaña de en atención el técnico de almacén dispone de las siguientes opciones en su interfaz gráfica: atender el pedido seleccionado, cancelar el pedido seleccionado, pasar un pedido determinado tanto si está completo como si está completo a envío, y salir a la interfaz de identificación.

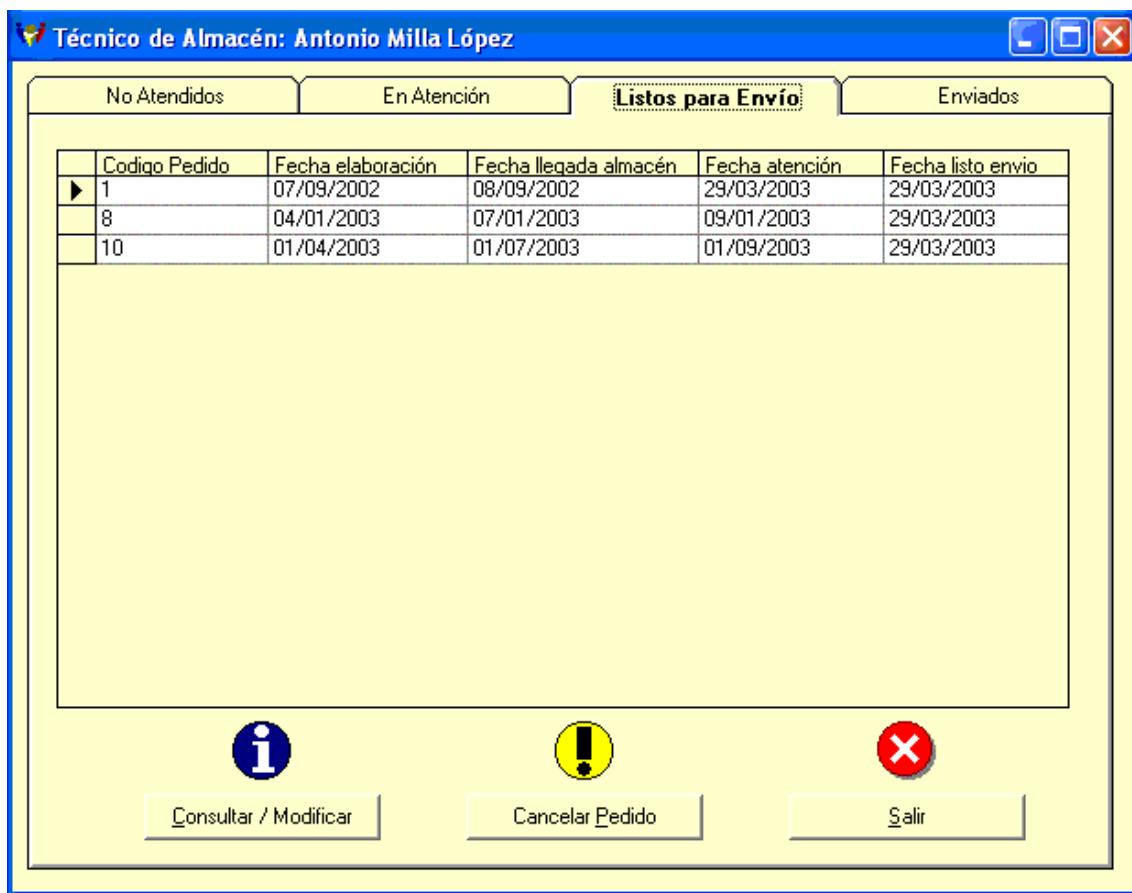
La interfaz de en atención es el siguiente:



En la pestaña de listos para envío, el técnico de almacén dispone de las siguientes opciones en su interfaz gráfica: consultar los detalles del pedido seleccionado, cancelar el pedido seleccionado o salir a la interfaz de identificación.

La interfaz de listos para envío es el siguiente:

PROYECTO FINAL TC1/TD



Dentro de la interfaz en de la consulta de pedidos no atendidos que se puede realizar desde la pestaña de no atendidos, se observa la siguiente interfaz gráfica:

? Consultar Pedidos no Atendidos

Datos del Pedido		Fecha de llegada al almacén	
Código pedido	Fecha elaboración		
3	10/11/2002	10/11/2002	
Dirección Envío	Nº	Pta	
C/ Melias	3	25	
Provincia	Localidad	CP	País
Valencia	Manises	46985	España

Productos

	Referencia	Nombre	Cantidad
▶	2	Mochila	2

Al atender, tanto si es la primera atención como si se trata de una modificación posterior de un pedido, se observa la siguiente interfaz gráfica, que dispone de las opciones siguientes: asignar cantidad al hacer doble clic sobre una línea de pedido, guardar los cambios realizados pulsando el botón guardar, pasar el pedido a envíos, generar una incidencia respecto a este envío o volver a la interfaz anterior pulsando el botón salir.

La interfaz de atender pedido es el siguiente:

PROYECTO FINAL TC1/TD

Atender Pedido [Modificación]

Datos del Técnico de Almacén

Código	Nombre
26485963M	Antonio Milla López

Datos del Pedido

Código del Pedido	Fecha llegada almacén	Fecha de atención	País	Provincia	
9	09/08/2002	29/03/2003	España	Valencia	
Dirección Envío		Nº	Pta	Localidad	CP
C/ Melias		3	25	Manises	46985

Productos

Cód. Artículo	Nombre	Cant. Solicitada	Cant. Asignada	Stock Real	Stock Disponible	Stock Asignado
1	Zapatillas	1	1	1000	998	2
3	Sudadera	2	0	6	6	0

Por último, al hacer doble clic sobre una línea de pedido, la interfaz gráfica que surge es:

Asignar Stock

Stock Artículo

Stock Real	Stock Disponible	Stock Asignado
1000	998	2

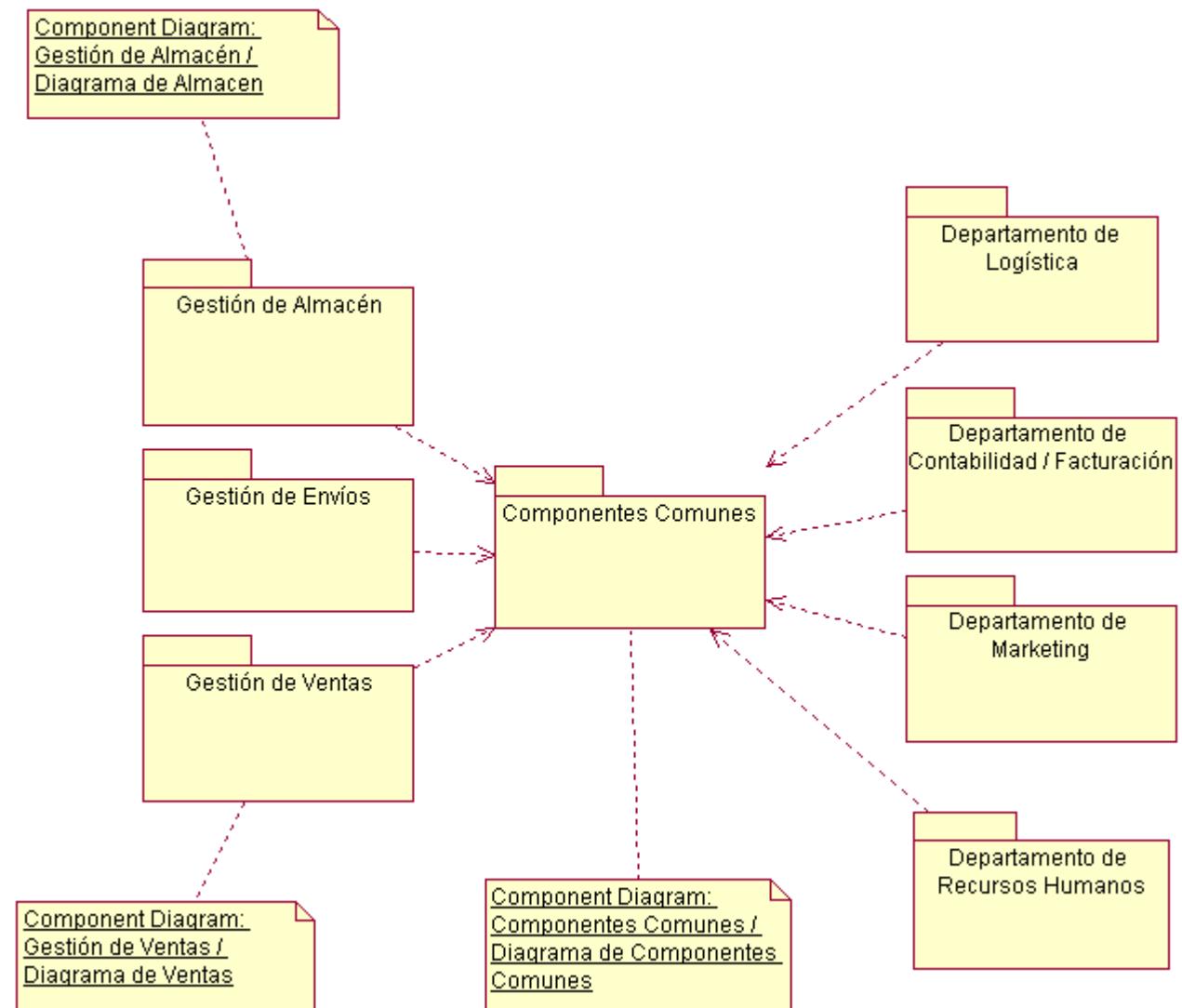
Editar línea

Código Artículo	Nombre	Cant. Solicitada	Cantidad a Asignar
1	Zapatillas	1	1

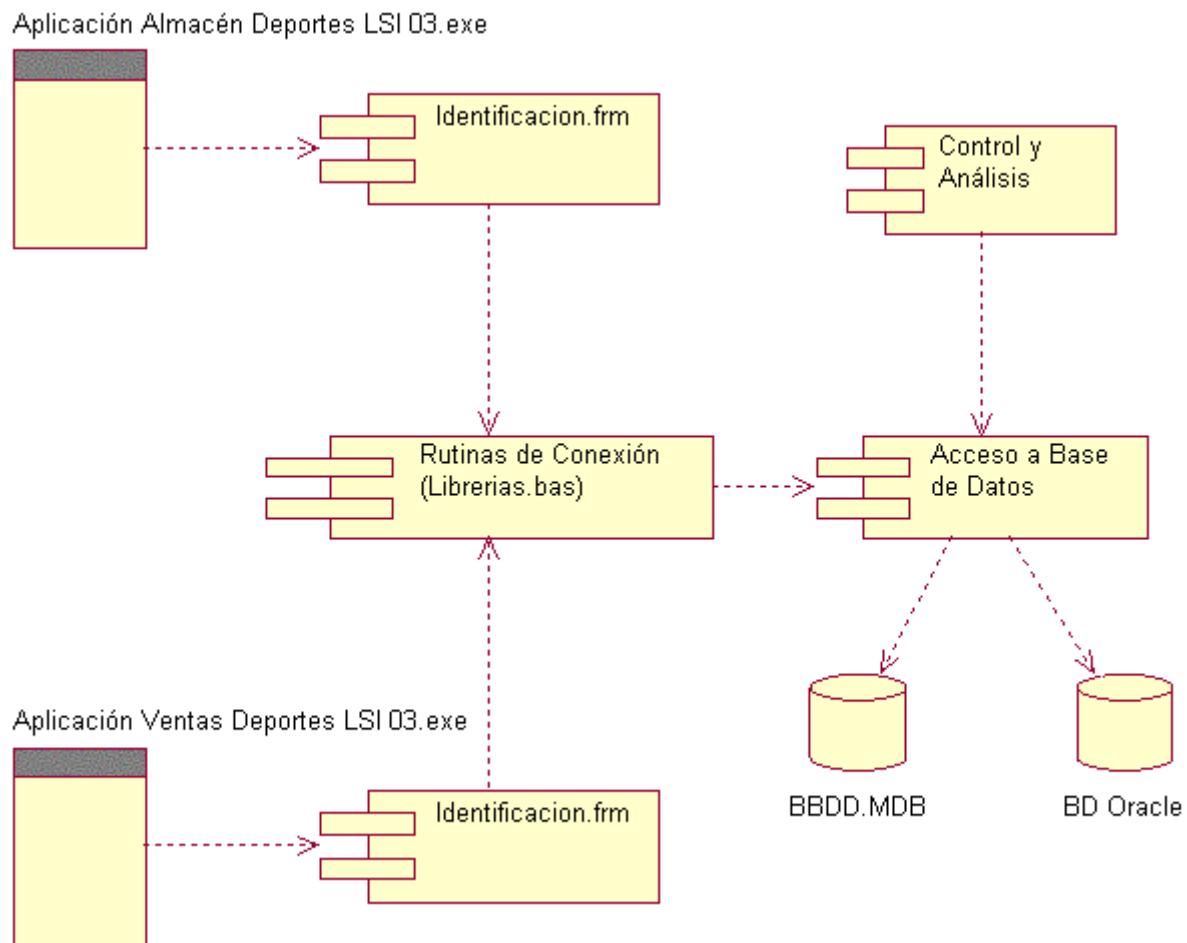
 

5.6.3. Diagrama de Componentes

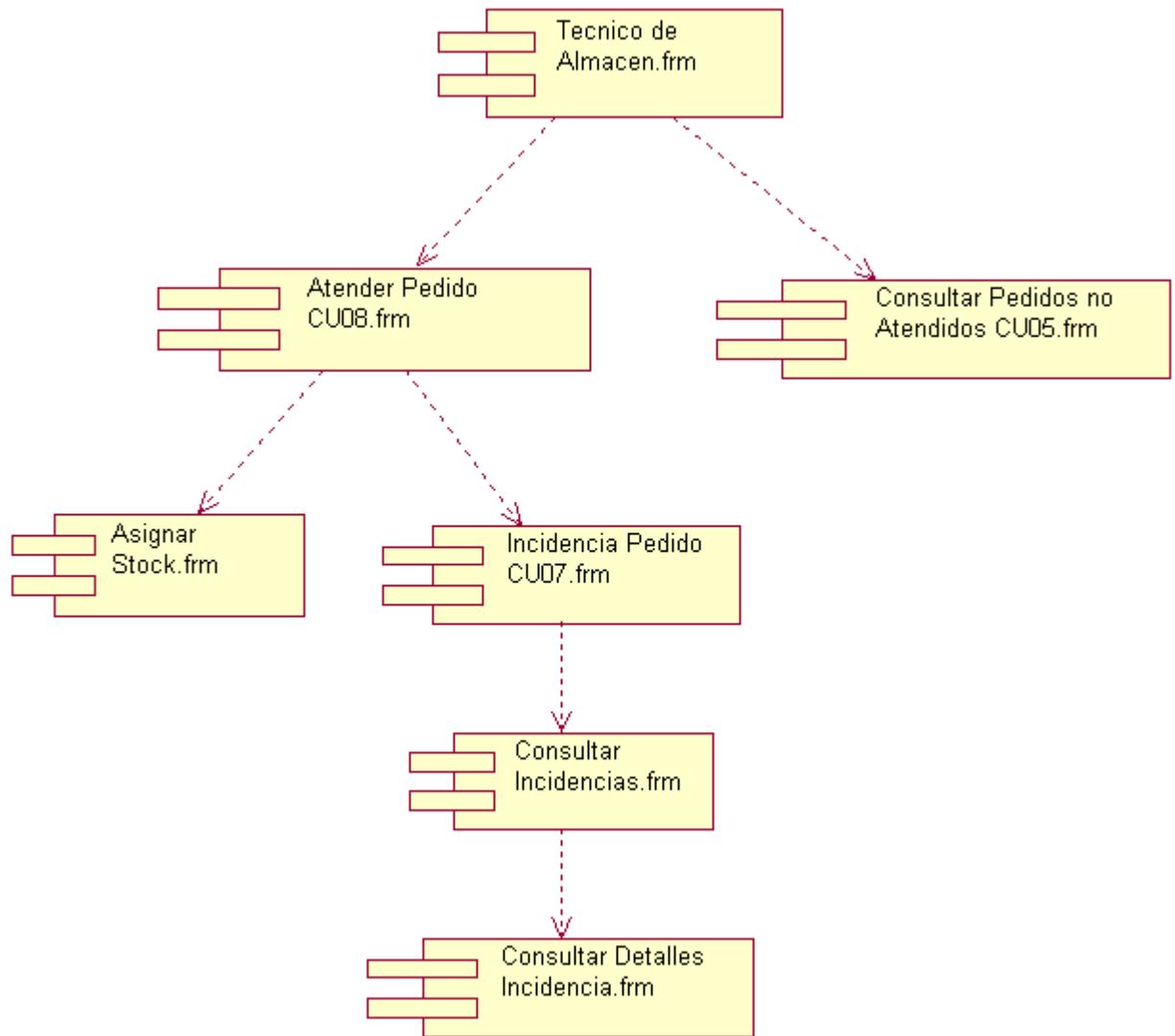
5.6.3.1. Diagrama Global de Paquetes



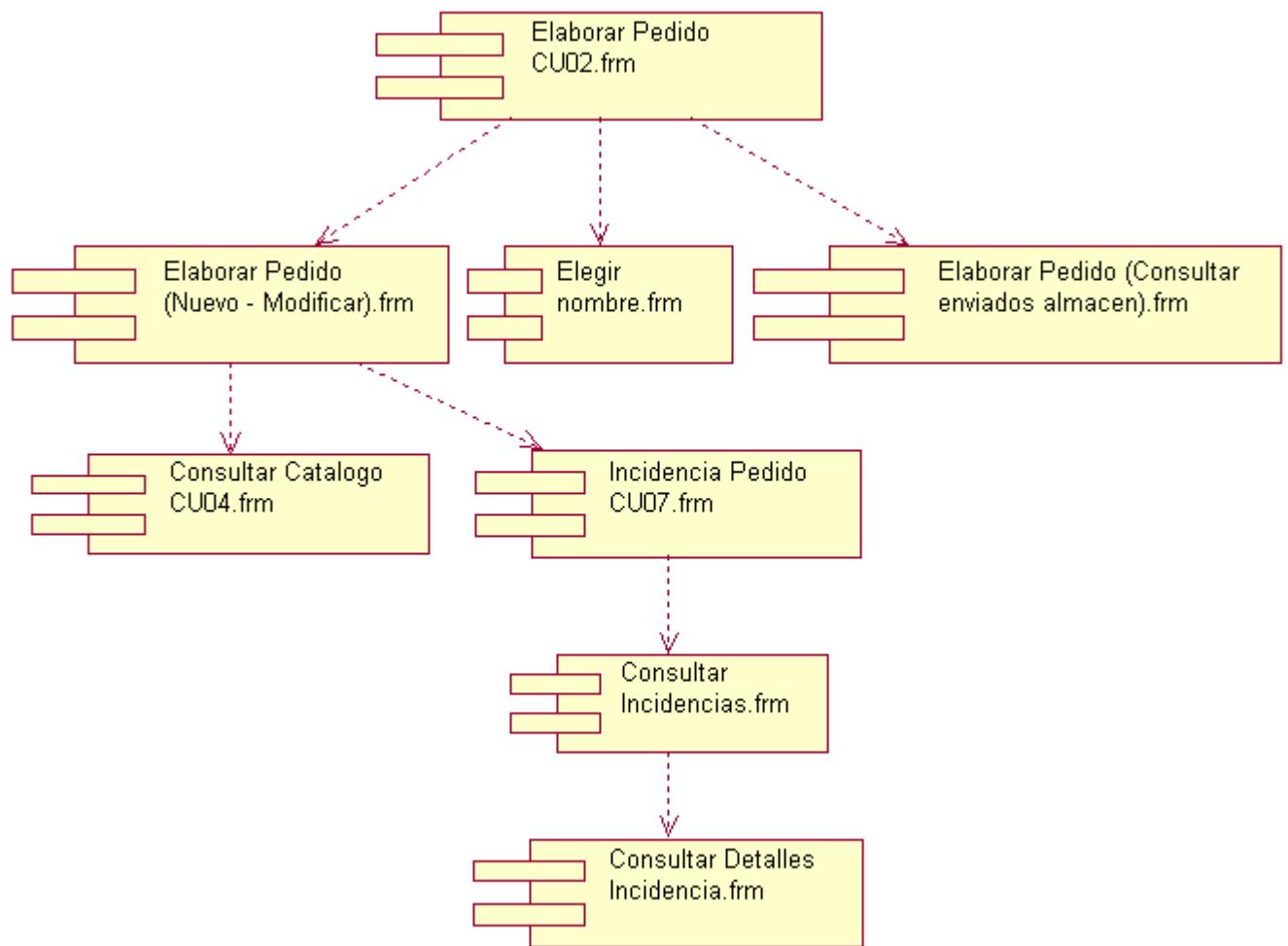
5.6.3.2. Diagrama de Componentes Comunes



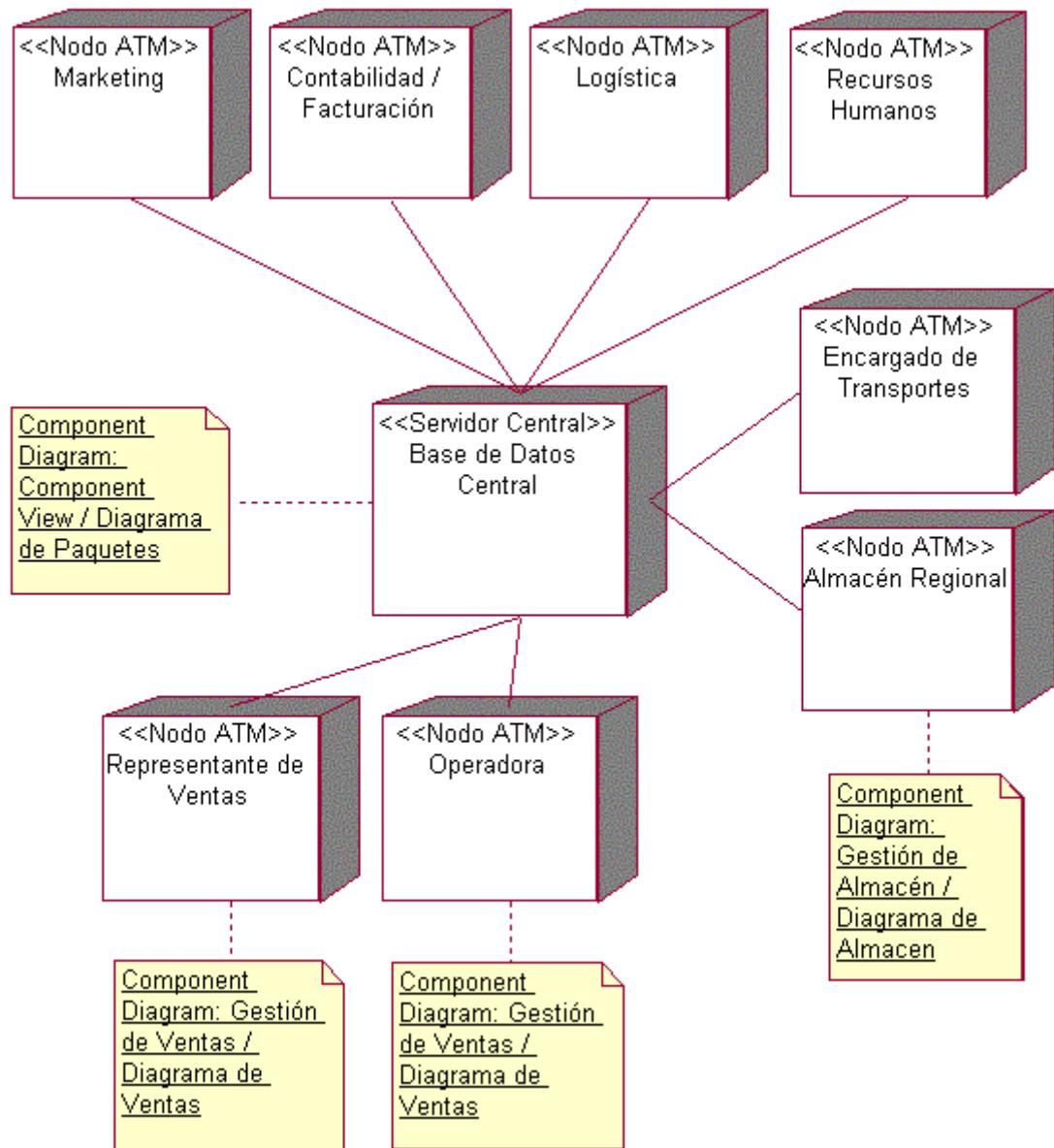
5.6.3.3. Diagrama de Componentes de Almacén



5.6.3.4. Diagrama de Componentes de Ventas



5.6.4. Diagrama de Despliegue



6. Plantillas de Artefactos del Proceso Unificado

A continuación se enumeran una serie de plantillas para confeccionar los diferentes artefactos que el Proceso Unificado recomienda.

Artefacto	Observaciones
Plan de Desarrollo de Software	
Visión	
Especificación de Caso de Uso	
Especificaciones Adicionales	
Glosario	
Caso de Prueba	
Registro de Tiempos	
Solicitud de Cambio	

7. Referencias y lectura recomendada

- [JBR00] Jacobson, I., Booch, G., Rumbaugh J., El Proceso Unificado de Desarrollo de Software, 2000 Addison Wesley
- [KRU00] Kruchten, P., The Rational Unified Process: An Introduction, 2000 Addison Wesley
- [KRU95] Kruchten, P. Architectural Blueprints—The “4+1” View Model of Software Architecture. IEEE Software 12 (6), Noviembre 1995, pp. 42-50.
- [RSC02] Rational Software Corporation, Product: Rational Software Corporation, 2002
- [RSC98] Rational Software Corporation, Rational Unified Process. Best Practices for Software Development Teams, 1998
- [BOO98] Booch, G. “Análisis y diseño orientado a objetos”. . 2da. Edición. Addison-Wesley. 1998
- [ELM00] ELMASRI, R. y NAVATHE, S. “Sistemas de bases de datos”. 2da. Edición. Addison-Wesley. 2000
- [LAU02] LAUDON, K., LAUDON, J. “Sistemas de Información Gerencial” 6ta. Edición. Pearson Educación. 2002
- [PRE02] PRESSMAN, R. “Ingeniería del Software. Un enfoque práctico”. 5ta edición. Mc Graw-Hill Interamericana. 2002
- [RJB00] RUMBAUGH, J., JACOBSON, I., BOOCH, G. “El lenguaje unificado de modelado. Manual de referencia”. Addison-Wesley. 2000

<http://www.microsoft.com/spanish/msdn>