

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

MODELADO DE PROCESOS PARA SU APLICACIÓN EN LA PRÁCTICA DE NEGOCIOS

T E S I S

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN COMPUTACIÓN

P R E S E N T A:

ODETTE MARGARITA BUSTAMANTE VALLÍN



DIRECTOR DE TESIS:

M. I. AURELIO SÁNCHEZ VACA

MÉXICO, D. F. **MARZO DEL 2006**

A mis padres

A Dios por darme todo lo que soy y todo lo que tengo.
A mis papás por su esfuerzo, por ser mi ejemplo, por guiarme, apoyarme y ser mi todo… no hay palabras para decir lo grande que son. ¡¡Los quiero!!
A mis hermanos y cuñados, por todo su apoyo, aguantarme y enseñarme. Los admiro y quiero, son un gran ejemplo a seguir.
A mis sobrinos, por ser mi alegría, motivación y por enseñarme tantas cosas. Me encantan… ¿por que están tan preciosos?
A Oscar por apoyarme, aconsejarme, acompañarme y por todo tu amor. Te quiero.
A mis padrinos Toña y Polo, porque se que de alguna forma siguen aquí apoyando a la familia.
A mis abuelitos por abrir las puertas de su casa.
A mis tías por su compañía y cariño.
A mis amigos, por el tiempo compartido.
Al M.I. Aurelio Sánchez Vaca por aceptar ser parte de este trabajo, por su asesoría y correcciones.
A la UNAM y a todos mis profesores, por toda su enseñanza… “Por mi raza hablará el espíritu”.

“Tu día pronto llegará… tu sueño vuelo emprenderá”

ÍNDICE

INTRODUCCIÓN..... 2
OBJETIVOS..... 2
PERSPECTIVA GENERAL DE LA TESIS..... 3
HACIA DONDE VAMOS..... 3

CAPÍTULO I. Por qué la administración de procesos..... 5
I.1. Que es un proceso..... 5
I.1.1. Procesos de negocio..... 5
I.2. Requisitos básicos de un proceso..... 6
I.3. Métodos para la identificación de un proceso..... 9
I.3.1. Método estructurado..... 9
I.3.2. Método creativo..... 10
I.3.3. Selección del método..... 11

CAPÍTULO II. Definición de UML..... 13
II.1. Antecedentes..... 13
II.2. Definición..... 16
II.3. Aplicación en los procesos de Negocio..... 20

CAPÍTULO III. Análisis del proceso de ventas con UML..... 23
III.1. Análisis del proceso de ventas..... 23
III.2. Definición de requerimientos..... 29
III.3. Asignación de responsabilidades..... 30

CAPÍTULO IV. Modelado del proceso de ventas con diagramas en UML..... 34
IV.1. Modelado con diagramas de Caso de Uso..... 34
IV.1.1. Descripción..... 34
IV.2. Diagramas de Iteración..... 40
IV.2.1. Modelado con diagrama de secuencia..... 41
IV.2.1.1. Descripción..... 41
IV.3. Modelado con Diagrama de Clases..... 44
IV.3.1. Descripción de clase..... 44
IV.4. Modelado con Diagrama de Paquetes..... 47
IV.4.1. Descripción de paquete..... 47

CAPÍTULO V. Definición de la Arquitectura..... 51
V.1. Descripción de los componentes..... 51
V.2. Modelado con Diagramas de Despliegue..... 56

CAPÍTULO VI Modelado General de Procesos de Ventas para Empresas..... 59
VI.1. Descripción del modelo..... 59
VI.2. Aplicación del modelo..... 62

CONCLUSIONES..... 130

GLOSARIO

BIBLIOGRAFÍA

INTRODUCCIÓN

INTRODUCCIÓN

Actualmente, para todas las empresas, independientemente de su tamaño y de las actividades que realiza, todas ellas se enfrentan con mercados competitivos en los que lo más importante es llegar a tener la satisfacción de sus clientes, viéndose reflejado en sus actividades económicas como empresa.

La administración de procesos percibe la empresa como un sistema interrelacionado de procesos que contribuyen conjuntamente a incrementar la satisfacción del cliente, y ésta coexiste con la administración funcional, asignando responsables a los procesos clave, haciendo posible una administración interfuncional generadora de valor para el cliente y que, por tanto, procura su satisfacción. Determina qué procesos necesitan ser mejorados o rediseñados, establece prioridades y provee de un contexto para iniciar y mantener planes de mejora que permitan alcanzar objetivos establecidos. De tal manera que la eficiencia y la eficacia de una empresa será el reflejo de los procesos que en ella son implantados.

La incorporación de las nuevas tecnologías de la información permite redefinir los procesos alcanzando grados de eficiencia y eficacia inimaginables hace unos años. Las empresas que sean capaces de descubrir estas posibilidades e implantarlas correctamente, conseguirán ventajas competitivas, debido a la disminución de costos y el aumento de flexibilidad frente a los requerimientos de los clientes.

Las empresas que actualmente no cuentan con procesos definidos, tienen un gran problema, debido a que día a día van creciendo y con ello el número de sus clientes aumenta también, por lo que el manejo y atención a ellos se ve afectado.

El no tener procedimientos a seguir implica no tener procesos, por lo que las actividades de los empleados se vuelven cotidianas y repetitivas sin tener ningún patrón a seguir, y sin asegurar si esos actos hechos por el empleado son los correctos o lo que mejor conviene a la empresa, todo eso implica un alto riesgo para la empresa ya que no hay forma de llevar un control de lo que se hizo y de lo que se está haciendo.

Por lo anterior mencionado, las empresas se encuentran con la necesidad de mejorar sus prácticas de negocio, entre ellas las de venta, por esta razón es necesario hacer un análisis de las actividades que se llevan a cabo para mejorarlas, establecer procesos acorde a sus necesidades, que puedan ser medidos, para posibles mejoras y lograr mayor efectividad y desempeño en la empresa.

OBJETIVOS

El objetivo principal de esta tesis es proporcionar la información necesaria para el modelado de procesos de venta dentro de la empresa, que pueda ser de ayuda para establecerlos, mejorar la eficiencia y eficacia de éstos y con ello el de la empresa.

Se busca satisfacer las necesidades que tiene la empresa en cuanto a procesos de venta, que éstos se conviertan en actividades sencillas y enriquecedoras de las personas que lo realicen, viéndose beneficiado el cliente con mejor atención y que todo ello lleve a la empresa a obtener los beneficios económicos esperados.

PERSPECTIVA GENERAL DE LA TESIS

CAPÍTULO I. Por qué la administración de procesos, se presenta la información referente a la administración de procesos, los requisitos de un proceso y los métodos para su identificación.

CAPÍTULO II. Definición de UML, se presentan un panorama general de lo que es UML y su aplicación en los procesos de negocio.

CAPÍTULO III. Análisis del proceso de ventas con UML, en este capítulo se tendrá el caso de estudio, es decir, lo que se necesita saber, por lo que se hará el análisis del proceso, se definirán los requerimientos y se asignarán las responsabilidades.

CAPÍTULO IV. Modelado de proceso de ventas con diagramas en UML, se describirán y aplicarán los procesos de ventas con diagramas UML como son, casos de uso, diagrama de secuencia, diagrama de paquetes y diagramas de clases.

CAPÍTULO V. Definición de la arquitectura, se mostrará la arquitectura de distribución, es decir, la arquitectura de hardware (servidores, clientes, dispositivos, conectores, etc.) y la interacción entre ellos.

CAPÍTULO VI. Modelado general de procesos de negocios para empresas, se describirá el modelo y su aplicación en los procesos de venta, es decir, la validación de los ciclos de vida de objetos.

HACIA DONDE VAMOS

La elaboración de esta tesis utiliza UML Unified Modeling Language (Lenguaje Unificado de Modelado), por medio de este lenguaje de modelado visual se logra especificar, visualizar, construir y documentar componentes de un sistema de negocios. Se usa para entender, diseñar, configurar, mantener y controlar la información de un modelo de negocio. UML es un lenguaje de propósito general para el modelado orientado a objetos.

Se pretende que por medio de esta información la empresa pueda establecer un modelado de procesos de negocio y con ello pueda tener los siguientes beneficios:

- Mayor beneficio económico
- Mayor satisfacción del cliente
- Mayor satisfacción del personal
- Mayor conocimiento y control de procesos
- Mejor flujo de información y materiales
- Disminución de los tiempos de proceso del producto o servicio
- Mayor flexibilidad frente a las necesidades de los clientes

POR QUÉ LA ADMINISTRACIÓN DE PROCESOS

I. POR QUÉ LA ADMINISTRACIÓN DE PROCESOS

I.1. ¿Qué es un proceso?

Un proceso es un ordenamiento específico de recursos y actividades de trabajo interrelacionadas entre sí, a través del tiempo y del espacio, con un comienzo, un fin, entradas y salidas claramente identificados de materiales o información, con valor añadido: es decir, una estructura para la acción. Los recursos pueden incluir personal, finanzas, instalaciones, equipos, técnicas y métodos.

En otras palabras, es la manera en la que se hacen el conjunto de actividades en la organización.

I.1.1. Procesos de negocio

Un proceso de negocio es una colección de actividades diseñadas para producir una salida específica para un cliente o mercado particular. Implica un fuerte énfasis en 'cómo' se hace el trabajo en una organización, en contraposición al enfoque en 'qué' de producto.

Un proceso de negocios[1]:

1. Tiene un objetivo.
2. Tiene entradas específicas.
3. Tiene salidas específicas.
4. Emplea recursos.
5. Tiene muchas actividades que se desarrollan en algún orden.
6. Puede afectar a más de una unidad organizacional. Impacto organizacional horizontal.
7. Crea valores de algún tipo para el cliente. Éste puede ser interno o externo.

En la figura I.1 se muestra una representación de un proceso de negocio.

[1] http://www.sparxsystems.cl/business_process_model.htm

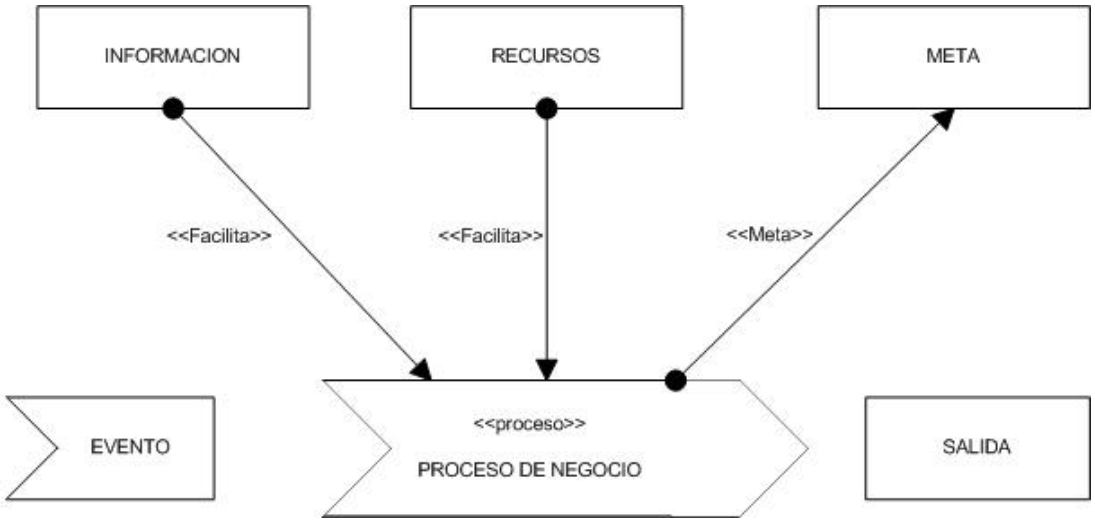


Figura I.1. Proceso de negocio

I.2. Requisitos básicos de un proceso

A continuación se muestra una lista de las características básicas de un proceso:

- Todos los procesos tienen que tener un responsable designado que asegure su cumplimiento y eficacia de manera continua.
- A cada proceso se le deberá asignar un nombre, el cual deberá ser representativo de lo que conceptualmente representa o se pretende representar.
- La totalidad de las actividades desarrolladas en al empresa deben estar incluidas en alguno de los procesos listados. De no ser así, deben tender a desaparecer.
- Será necesario establecer procesos clave, es decir, procesos relevantes según el impacto de los procesos relacionados con los objetivos estratégicos y las repercusiones en los clientes.
- Se recomiendan que el número de procesos sea pequeño, debido a que el manejo de los procesos es más eficaz si estos son pocos.
- Todos los procesos tienen que ser capaces de satisfacer los ciclos P, D, C, A[2] mostrados en la figura I.2. Tienen que tener indicadores que permitan visualizar de forma gráfica la evolución de los mismos. Tienen que ser planificados en la fase P, tienen que asegurarse su cumplimiento en la fase D, tienen que servir para realizar el seguimiento en la fase C y tiene que utilizarse en la fase A para ajustar y/o establecer objetivos.

[2] Concepto ideado originalmente por Shewhart, también conocido como ciclo Deming, después de que éste lo llevó a Japón en 1950.

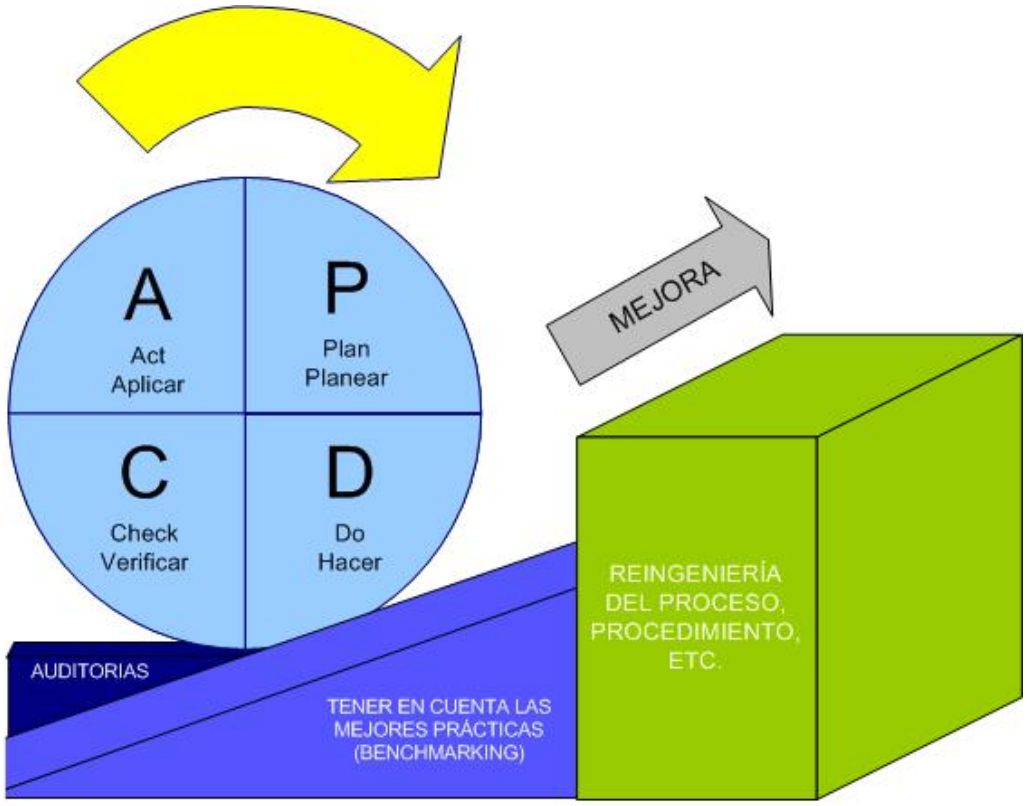


Figura I.2. Ciclo PDCA

Donde:

P: Plan => Planear, es la etapa en la cual se programan y planifican las actividades que se van a realizar

D: Do => Hacer, en esta etapa se implantan y ejecutan las actividades propuestas.

C: Check => Verificar, en esta etapa se comprueban y verifican si las actividades se han resuelto bien y si los resultados obtenidos son de acuerdo a los objetivos que se tienen.

A: Act => En esta etapa se aplican los resultados obtenidos para analizar y continuar estudiando posibles mejoras de acuerdo a los objetivos o incluso para reajustar éstos.

- Es necesario establecer los límites de los procesos, identificando las entradas y salidas, recogiendo los clientes y proveedores del proceso, así como aquellos otros procesos de la empresa que tienen alguna relación.

- Dentro de los procesos habrá que distinguir y documentar las actividades y subprocesos relacionados.

- Los subprocesos también tienen que garantizar que se cumplen los ciclos P, D, C, A comentados anteriormente.

- Es necesario establecer indicadores, los cuales son necesarios para poder mejorar, ya que con estos podemos interpretar lo que está ocurriendo, tomar medidas cuando las variables se salen de los límites establecidos, definir la necesidad de introducir un cambio y evaluar sus consecuencias y planificar las actividades para dar respuesta a nuevas necesidades.

- Es recomendable planificar y realizar periódicamente una reingeniería de los procesos de administración para alcanzar mejoras notables en determinados parámetros como costos, calidad, servicio y rapidez de respuesta.

Una forma más moderna y completa de ver estos ciclos de revisión y mejora se encuentra dentro de la filosofía REDER[3], que es un esquema lógico el cuál está formado por cuatro elementos, los cuales se muestran en la figura I.3:

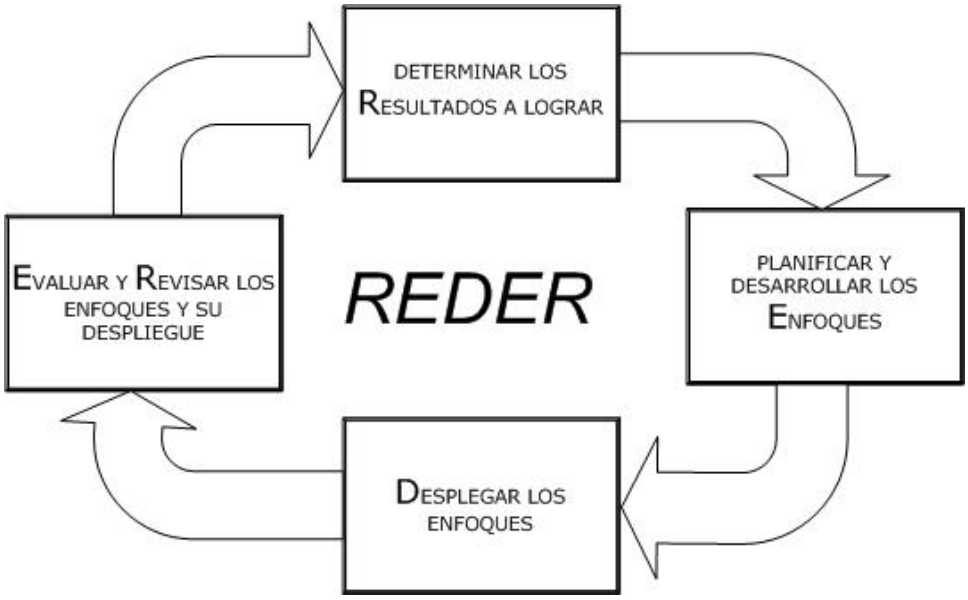


Figura I.3. La filosofía REDER

El esquema lógico establece lo que una organización necesita realizar:

- Determinar los *resultados* que quiere lograr como parte del proceso de elaboración de su política y estrategia. Estos resultados cubren el rendimiento de la organización, tanto en términos económicos y financieros como operativos, así como las percepciones de todos los grupos de interés de la organización.

[3] http://web.jet.es/amozarrain/reder.htm

- Planificar y desarrollar una serie de *enfoques* solidamente fundamentados e integrados que la lleven a obtener los resultados requeridos ahora y en el futuro.

- *Desplegar* los enfoques de manera sistemática para asegurar una implantación completa.

- *Evaluar y revisar* los enfoques utilizados basándose en el seguimiento y análisis de los resultados alcanzados y en las actividades continuas de aprendizaje. En función de todo ello, identificar, establecer prioridades, planificar e implantar las mejoras que sean necesarias.

A continuación se describen con detalle los elementos del concepto REDER que deben abordarse:

Enfoque: Este abarca lo que una organización ha planificado hacer y las razones para ello. En una organización excelente, el enfoque estará, solidamente fundamentado, es decir, tendrá una lógica clara, procesos bien definidos y desarrollados y una clara orientación hacia las necesidades de todos los grupos de interés; y, por otra, estará integrado, es decir, apoyara la política y estrategia y, cuando así convenga, estará vinculado a otros enfoques.

Despliegue: Se ocupa de lo que hace una organización para desplegar el enfoque. En una organización considerada excelente, el enfoque se implantara en las áreas relevantes y de un modo sistemático.

Evaluación y revisión: Aquí se aborda lo que hace una organización para evaluar y revisar el enfoque y el despliegue de dicho enfoque. En una organización considerada excelente, el enfoque y su despliegue estarán sujetos a mediciones regulares y se realizarán actividades de aprendizaje, empleándose el resultado de ello en identificar, establecer prioridades, planificar e implantar la mejora.

Resultados: Este elemento se ocupa de los logros alcanzados por una organización considerada excelente, mostrará tendencias positivas y/o un buen rendimiento sostenido, los objetivos serán adecuados y se alcanzarán, y el rendimiento será bueno comparado con el de otras organizaciones y será consecuencia de los enfoques. Además, el ámbito de aplicación de los resultados abordará las áreas relevantes.

I.3. Métodos para la identificación de un proceso

Se puede asegurar que existen muchos métodos para la identificación de los procesos. Pero éstos se pueden englobar en dos grandes grupos: Método estructurado y método creativo.

I.3.1. Método estructurado

En este grupo se pueden englobar todos aquellos sistemas básicamente complejos que sirven para la identificación de los procesos de administración. Con este método se definen y establecen los pasos necesarios para garantizar que un

producto satisfaga a sus clientes. Lo que tienen en común todos estos sistemas, es que los mismos están diseñados por personas expertas. Normalmente su implantación requiere de algún tipo de asistencia externa.

Ventajas:

Son sistemas estructurados que sirven para identificar y documentar un proceso de administración. Se dan pautas, guías, soportes y hasta plantillas.

Con este método se puede tener un estándar en el desarrollo y control de los proyectos.

Estos sistemas permiten identificar áreas de administración que no se abordan y/o ineficientes. Los procesos y subprocesos relacionados están perfectamente documentados.

Si se consigue mantener actualizada toda la documentación asociada a los mismos se convierten en herramientas validas para la formación de los nuevos ingresos.

Inconvenientes:

Los procesos de administración están tan documentados que en algunas ocasiones más que ser una herramienta de administración operativa, son complejos en su manejo o lectura.

Los métodos informáticos requieren menos papel, pero para entenderlos e interpretarlos se requiere de una persona experta que por un lado conozca la herramienta y por otro lado domine la administración que está reflejada en los gráficos.

Otro de los problemas asociados a este tipo de sistemas es que normalmente no suelen saber que hacer con los procedimientos existentes y sus sistemas relacionados, de tal manera que cuando hay un nuevo sistema de procesos en la empresa, no saben como relacionarlo con los sistemas existentes.

I.3.2. Método creativo

Los seres humanos usamos la creatividad cada vez que requerimos resolver un problema o enfrentarnos a situaciones nuevas, es por ello que en este grupo se pueden englobar todos aquellos métodos que las empresas están ideando e implantado de forma interna, los cuales normalmente son motivados por las malas experiencias y/o por la ineficiencia del método anterior.

El método creativo está fuertemente orientado al trabajo en grupo pero también puede utilizarse en la solución de problemas. Cuando se enfoca al trabajo individual, el método creativo también se conoce como pensamiento horizontal. El método creativo se puede describir con los siguientes simples pasos:

1. Enunciación del problema

2. Enunciación de restricciones y de metas

3. Criterios de evaluación de propuestas de solución

4. Lluvia de Ideas de propuestas de solución

5. Revisión cruzada de las ideas (Sólo si es un equipo de trabajo)

6. Evaluación de las opciones

El resultado final del método creativo es una propuesta de solución que ha de implantarse.

Ventajas:

El sistema de administración esta mucho más integrado, ya que tanto el método ideado como todos los soportes relacionados están creados internamente por miembros de la organización. Estos soportes y métodos se convierten con poco esfuerzo en documentos "entendibles" por el resto del personal.

La documentación se reduce drásticamente. Los procedimientos desaparecen y se "convierten" y/o se incorporan a los procesos relacionados.

Inconvenientes:

Se requiere de personas expertas en todos los campos relacionados.

Se debe hacer mayor énfasis en la formación de las nuevas incorporaciones, ya que buena parte del conocimiento no esta ni en papel ni en soportes informáticos. En estos casos se acostumbra mucho la difusión del conocimiento de manera hablada.

I.3.3. Selección del método

Como bien se mencionó anteriormente, existen muchos métodos, y aunque se puede decir que los dos expuestos engloban de alguna manera a todos, la elección del método dependerá del conocimiento que tengan los miembros de la empresa y/o de la situación en el cual se encuentre la misma. Como se mencionó ambos métodos tienes ventajas y desventajas, sin embargo se puede decir que el aplicar método estructurado es mejor, puesto que en él la información queda impresa para futuros seguimientos o capacitaciones, siempre y cuando ésta sea clara y precisa.

DEFINICIÓN DE UML

II. DEFINICIÓN DE UML

II.1. Antecedentes

UML fue desarrollado en un esfuerzo para simplificar y consolidar el gran número de métodos de desarrollo orientado a objetos que habían surgido.

Los métodos de desarrollo para los lenguajes de programación tradicionales tales como Cobol y Fortran, emergieron en los años 70 y llegaron a ser ampliamente difundidos en los 80. Principalmente entre ellos estaba el análisis estructurado y el diseño estructurado y sus variantes tales como diseño estructurado de tiempo real y otros. Los resultados no fueron siempre tal y como se esperaban –muchos sistemas de ingeniería de software asistidos por computadora (CASE) fueron poco más que generadores de informes que extraían diseños después de que la implementación estuviera terminada- pero los métodos incluían buenas ideas que fueron usadas eficientemente en algunos casos para la construcción de grandes sistemas CASE y métodos de desarrollo. La mayoría de los negocios desarrollaban su software internamente según sus propias necesidades, sin la relación de enfrentamiento entre cliente y contratista.

El primer lenguaje que es generalmente reconocido como orientado a objetos es Simula 67, desarrollado en 1967. Este lenguaje nunca tuvo un significativo seguimiento, aunque influyó notablemente en los desarrolladores de varios de los lenguajes orientados a objetos posteriores. El movimiento de la orientación a objetos se convirtió en activo con la amplia difusión de la disponibilidad del Smalltalk a principios de los 80, seguido por otros lenguajes orientados a objetos como Objetive C, C++, Eiffel, y CLOS. El uso real de los lenguajes orientados a objetos fue limitado al principio, pero la orientación a objetos atrajo mucho la atención.

Hubo algunos intentos tempranos de unificar los conceptos entre métodos. Un ejemplo destacable fue Fusion por Coleman y sus colegas, el cual incluyó conceptos de OMT (Object Modeling Technique) y Booch. El primer intento exitoso de combinar y reemplazar los métodos existentes llegó cuando James Rumbaugh se unió a Grady Booch en Rational Software Corporation en octubre de 1994, marcándose de esta forma el inicio del desarrollo de UML.

Debido a que los métodos Booch y OMT ya habían madurado independientemente y eran reconocidos como métodos líderes en el desarrollo orientado a objetos, Booch y Rumbaugh unieron fuerzas para forjar una unificación completa de los dos métodos. Una versión preliminar 0.8 del “método unificado” fue dada conocer en octubre de 1995. Poco después, Ivar Jacobson, creador del método OOSE (Object Oriented Software Engineering) y, sobre todo, del ingenioso concepto *use case* (caso de uso), unió su compañía Objectory a Rational y a su trabajo de unificación, uniendo con esto el método OOSE, de lo cual surgió la versión 0.9 de UML, en junio de 1996. El Nombre de Objectory es ahora dado mayormente para describir el proceso que acompaña al UML el “Rational Unified Process”. Durante 1996 se realizan sucesivas modificaciones en base a aportes de muchas otras personas de las cuales surgió la versión 0.91.

En 1996, el Object Management Group (OMG), un pilar estándar para la comunidad del diseño orientado a objetos, creado en 1989 responsable de la creación, desarrollo y revisión de especificaciones para la industria del software, publicó una petición con propósito de un metamodelo orientado a objetos de semántica y

notación estándares. UML, en su versión 1.0, fue propuesto como una respuesta a esta petición en enero de 1997. Hubo otras cinco propuestas rivales. Durante el transcurso de 1997, los seis promotores de las propuestas, unieron su trabajo y presentaron al OMG un documento revisado de UML, llamado UML versión 1.1. El Lenguaje Unificado de Modelado fue adoptado unánimemente por los miembros de OMG como estándar en noviembre de 1997. El OMG llama a este documento OMG UML versión 1.1, y la versión 1.2 salió en junio de 1998. Basado en las experiencias de las empresas participantes en UML 1.0, UML 1.1 y UML 1.2, en junio de 1999 surge UML versión 1.3. En Septiembre de 2001 se publicó la especificación de la versión 1.4. Desde aquella versión han habido varias revisiones que gestiona la OMG Revision Task Force. La última versión aprobada es la UML 2.0 Superstructure, publicado en agosto de 2003. En estos momentos se está desarrollando actualizaciones a esta versión en la que se incluirán cambios importantes (principalmente para añadir nuevos diagramas).[4]

En la figura II.1 se muestra de manera gráfica la evolución a través del tiempo de UML con sus diferentes versiones.

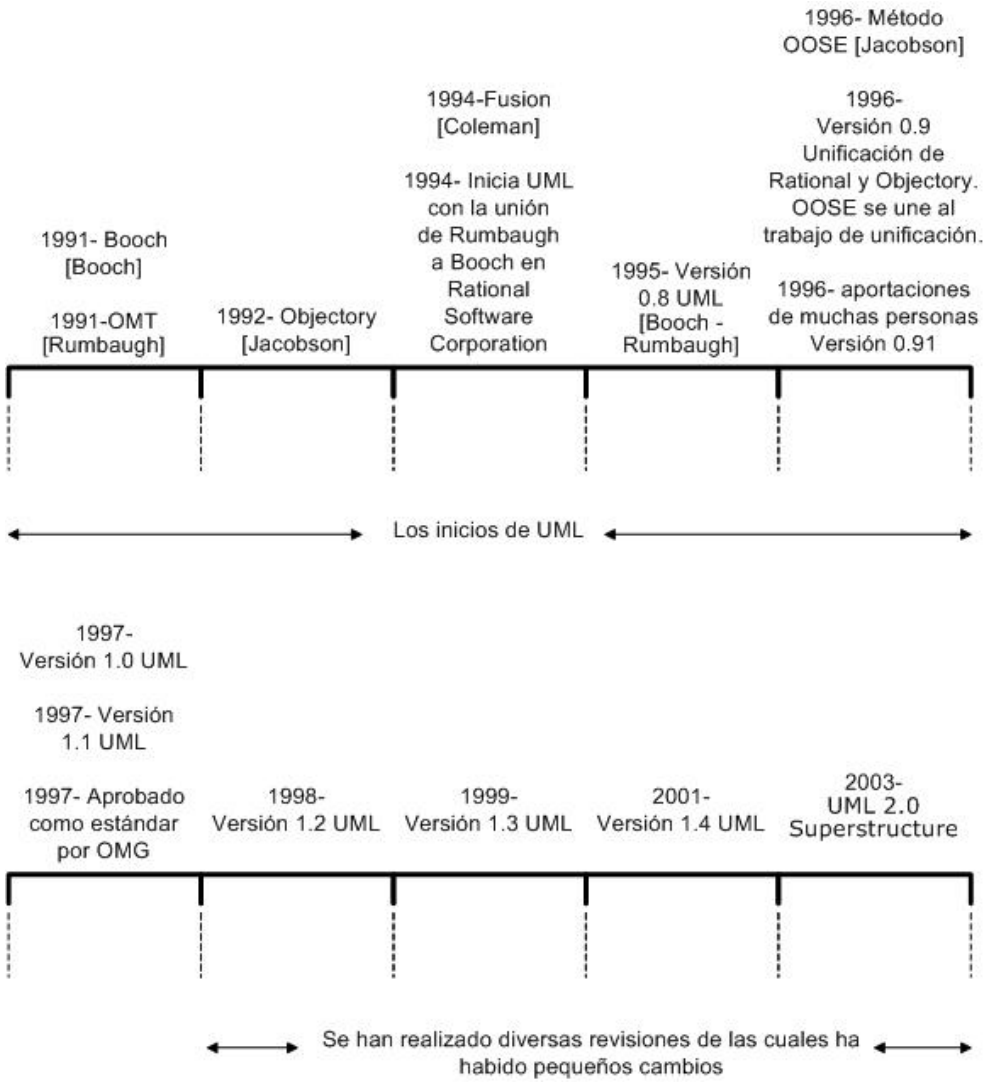


Figura II.1. Evolución de UML

[4] El lenguaje unificado de modelado. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 1999.

Un metamodelo es la descripción de un modelo. Un metamodelo procura hacer un lenguaje exacto, definiendo su semántica, pero hay una tensión para permitir las extensiones para las nuevas situaciones. Un metamodelo describe el contenido de un lenguaje bien formado, mientras que un lenguaje de programación describe un programa bien formado.

Es importante mencionar que la palabra unificado tiene varios significados relevantes para UML, y es por los cuales UML adopta ese nombre. Estos significados son los siguientes[5]:

- *A través de los métodos históricos y notaciones.* UML es la combinación de conceptos que son aceptados por muchos métodos orientados a objetos, para los cuales tiene una definición clara, una notación y una terminología. Por lo que UML puede representar la mayoría de los modelos existentes de la misma forma o mejor como son originalmente.

- *A través del ciclo de vida de desarrollo*. UML no tiene saltos ni discontinuidades desde los requisitos hasta la implantación. Por tal motivo se pueden utilizar los mismos conceptos y notación en las diferentes etapas de desarrollo.

- *A través de los dominios de aplicación*. UML está pensado para modelar la mayoría de los dominios de aplicación, incluyendo sistemas grandes, complejos, de tiempo real, distribuidos, con tratamiento intensivo de datos o con cálculo intensivo entre otras propiedades.

- *A través de los lenguajes de implementación y plataformas.* UML está pensado para ser usado en sistemas desarrollados en varios lenguajes de implementación y plataformas, incluyendo lenguajes de programación, bases de datos, documentos de organización, firmware y otros más.

- *A través de procesos de desarrollo.* El UML es un lenguaje, no un proceso de desarrollo detallado. Por lo que se pretende que sea usado como lenguaje de modelado a los procesos de desarrollo, al igual que un lenguaje de programación de propósito general puede ser usado en varios estilos de programación. UML es pensado para apoyar un estilo de desarrollo iterativo e incremental.

- *A través de los conceptos internos.* Uno de los resultados de la construcción del metamodelo de UML fue, que sus creadores descubrieron y representaron las relaciones subyacentes entre varios conceptos, para que de esta manera se puedan tener conceptos de modelado de manera abierta, aplicable a muchas situaciones conocidas y desconocidas.

UML se ha diseñado realizando combinaciones de una gran cantidad de estándares, aunque se rige a través de tres metodologías procedentes de la colaboración de los tres creadores de UML ya citados, J. Rumbaugh, G. Booch e I. Jacobson, así como del análisis y estudio de alrededor de 20 métodos estándares que a su vez se han integrado en otro estándar, en este caso, UML; esta fue una gran iniciativa de los tres creadores que pusieron las especificaciones de UML a la consideración de la comunidad informática mundial, antes de su publicación. UML es un lenguaje para modelar, que es el procedimiento que emplean los ingenieros para el diseño de software antes de pasar a su construcción, al igual que sucede con cualquier producto manufacturado o fabricado en serie.

[5] El lenguaje unificado de modelado. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 1999.

Hubo varios objetivos detrás del desarrollo de UML. El primero y más importante, UML es un lenguaje de modelado de propósito general que pueden usar todos los modeladores. No tiene propietario y está basado en el común acuerdo de gran parte de la comunidad informática. Está pensado para reemplazar al menos los modelos de OMT, Booch y Objectory, así como aquéllos de otros participantes de la propuesta. Se pensó para ser tan familiar como sea posible, usar la notación de OMT, Booch, Objectory y otros métodos importantes. Pretende abordar los problemas actuales del desarrollo de software, tales como gran tamaño, distribución, concurrencia, patrones, y desarrollo en equipo.

UML necesita ser lo suficientemente expresivo para manejar todos los conceptos que se originan en un sistema moderno, tales como la concurrencia y distribución, así como también los mecanismos de la ingeniería de software, tales como encapsulación y componentes. Debe ser un lenguaje universal, como cualquier lenguaje de programación de propósito general.

UML ayuda al usuario a entender la realidad de la tecnología y la posibilidad de que se reflexione antes de invertir y gastar grandes cantidades en proyectos que no estén seguros en su desarrollo, reduciendo el costo y el tiempo empleado en la construcción de las piezas que constituirán el modelo.

Sin embargo, desde el punto de vista puramente tecnológico, UML tiene una gran cantidad de propiedades que han sido las que, realmente, han contribuido a hacer realmente de UML un estándar efectivo de la industria. Algunas de las propiedades de UML como lenguaje de modelado estándar son:

- Concurrencia, es un lenguaje distribuido y adecuado a las necesidades de conectividad actuales y futuras.
- Ampliamente utilizado por la industria desde su adopción por OMG.
- Reemplaza a decenas de notaciones empleadas con otros lenguajes.
- Modela estructuras complejas.
- Las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.
- Emplea operaciones abstractas como guía para variaciones futuras, añadiendo variables si es necesario.
- Comportamiento del sistema: casos de uso, diagramas de secuencia y de colaboraciones, que sirven para evaluar el estado de las máquinas.

II.2. Definición

UML Unified Modeling Language (Lenguaje Unificado de Modelado), es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar componentes de un sistema de software. Se usa para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir. UML Es un lenguaje de propósito general para el modelado orientado a objetos. Representa

una colección de las mejores prácticas desde el punto de vista de la ingeniería que ha resultado exitoso en el modelado de sistemas complejos de gran porte.[6]

Un modelo es una representación en cierto medio, de algo en el mismo u otro medio. El modelo capta los aspectos importantes de lo que estamos modelando, desde cierto punto de vista, y simplifica u omite el resto.

Los modelos se usan para muchos propósitos. Para captar y enumerar exhaustivamente los requisitos y el dominio de conocimiento, de forma que todos los implicados pueden entenderlos y estar de acuerdo con ellos.

Un modelo es una abstracción de un sistema físico; mediante diagramas se logra una presentación gráfica de una colección de elementos pertenecientes al modelo. UML soporta los siguientes tipos de diagramas: de clases, de objetos, de casos de uso, de secuencia, de colaboración, de estados, de actividad, de componentes y de despliegue. Un diagrama típicamente no esta lo suficientemente refinado como para proveer todos los aspectos relevantes de una especificación; existe, entre otras cosas, una necesidad de describir restricciones adicionales sobre los elementos de un modelo.

Los modelos tienen dos aspectos importantes: Información semántica (semántica) y presentación visual (notación).

El aspecto semántico capta el significado de una aplicación como una red de construcciones lógicas, por ejemplo clases, asociaciones, estados, casos de uso, y mensajes. Los elementos semánticos del modelo llevan el significado del modelo, es decir, transportan la semántica. Los elementos semánticos del modelo se utilizan para la generación del código, la comprobación de la validez, las métricas de complejidad, etc. La información semántica a menudo es llamada *el modelo*.

La presentación visual muestra la información semántica de modo que pueda ser considerada, hojeada y corregida por los seres humanos. Los elementos de la presentación llevan la presentación visual del modelo. No agregan significado, sino que organizan la presentación, para acentuar la organización del modelo de una manera útil.

UML es un lenguaje para hacer modelos y es independiente de los métodos de análisis y diseño. Existen diferencias importantes entre un método y un lenguaje de modelado. Un método es una manera explícita de estructurar el pensamiento y las acciones de cada individuo. Además, el método le dice al usuario qué hacer, cómo hacerlo, cuándo hacerlo y por qué hacerlo; mientras que el lenguaje de modelado carece de estas instrucciones. Los métodos contienen modelos y esos modelos son utilizados para describir algo y comunicar los resultados del uso del método.

Un modelo es expresado en un lenguaje de modelado. Un lenguaje de modelado consiste de vistas, diagramas, elementos de modelo (los símbolos utilizados en los modelos) y un conjunto de mecanismos generales o reglas que indican cómo utilizar los elementos. Las reglas son sintácticas, semánticas y pragmáticas.

En la figura II.2 se muestran los componentes de un lenguaje de modelado.

[6] El lenguaje unificado de modelado. Manual de referencia. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 2000.

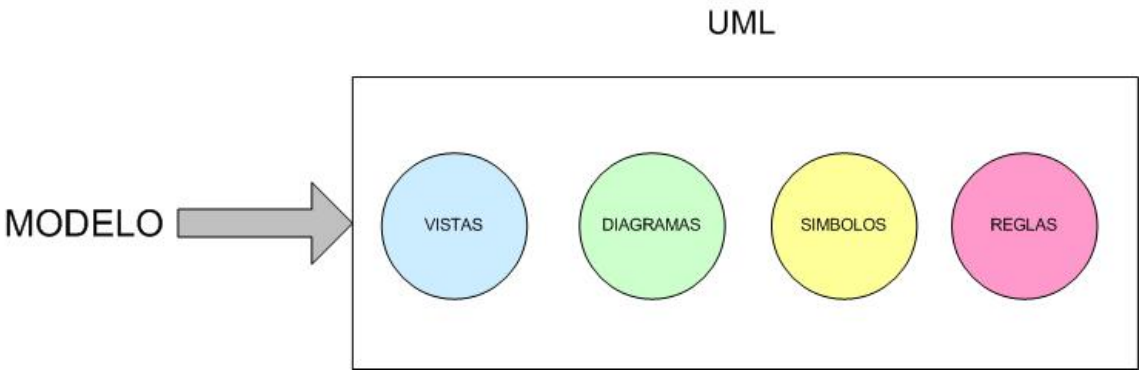


Figura II.2. Lenguaje de modelado

Vistas: Las vistas muestran diferentes aspectos del sistema modelado. Una vista no es una gráfica, pero sí una abstracción que consiste en un número de diagramas y todos esos diagramas juntos muestran una "fotografía" completa del sistema. Las vistas también ligan el lenguaje de modelado a los métodos o procesos elegidos para el desarrollo.

Una vista es simplemente un subconjunto de UML que modela construcciones que representan un aspecto de un sistema. Las vistas se pueden dividir en tres áreas: clasificación estructural, comportamiento dinámico, y gestión del modelo.[7]

La clasificación estructural describe los elementos del sistema y sus relaciones con otros elementos. Los clasificadores incluyen clases, casos de uso, componentes, y nodos, estos elementos proporcionan la base sobre la cual se construye el comportamiento dinámico. La clasificación de las vistas incluye la vista estática, la vista de casos de uso y la vista de implementación.

El comportamiento dinámico describe el comportamiento de un sistema en el tiempo. El comportamiento se puede describir como serie de cambios a las fotos del sistema dibujadas a partir de la visión estática. Las vistas de comportamiento dinámico incluyen vista de la máquina de estados, la vista de actividad, y la vista de interacción.

La gestión del modelo modela la organización del modelo en sí mismo. Un modelo abarca un conjunto de paquetes que contienen los elementos del modelo, tales como clases, máquinas de estados y casos de uso. Los paquetes son unidades para manipular el contenido de un modelo, así como unidades para el control de acceso y el control de configuración. La vista de gestión del modelo se representa en diagramas de paquetes.

- *Vista estática:* La vista estática modela los conceptos del dominio de la aplicación, así como los conceptos internos inventados como parte de la implementación de la aplicación. Esta visión es estática por que no describe el comportamiento del sistema dependiente del tiempo, que se describe en otras vistas. Los componentes principales de la vista estática son las clases y sus relaciones: asociación, generalización, y varias clases de dependencia.

[7] El lenguaje unificado de modelado. Manual de referencia. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 2000.

- *Vista de los casos de uso*: La vista de los casos de uso modela la funcionalidad del sistema según lo perciben los usuarios externos, llamados actores. El propósito de la vista de casos de uso es enumerar a los actores y los casos de uso, y demostrar que actores participan en cada caso de uso.

- *Vista de interacción*: La vista de interacción describe secuencias de intercambios de mensajes entre los roles que implementan el comportamiento de un sistema. Esta visión proporciona una vista integral del comportamiento de un sistema, es decir muestra el flujo de control a través de muchos objetos. La vista de interacción se exhibe en dos diagramas centrados en distintos aspectos: diagramas de secuencia y diagramas de colaboración.

- *Vista de la máquina de estados*: Una máquina de estados modela las posibles historias de vida de un objeto de una clase. Una máquina de estados contiene los estados conectados por transiciones. Cada estado modela un pequeño período de tiempo, durante la vida de un objeto, en el que satisface ciertas condiciones. Las máquinas de estados se pueden utilizar para describir interfaces de usuario, controladores de dispositivo, y otros sistemas reactivos.

- *Vista de actividades*: Un diagrama de actividades es una variante de una máquina de estados que muestra las actividades de computación implicadas en la ejecución de un cálculo. Un estado de actividad representa una actividad: un paso en el flujo de trabajo o la ejecución de una operación.

Las vistas mencionadas anteriormente modelan los conceptos de la aplicación desde un punto de vista lógico. Las vistas físicas modelan la estructura de la implementación de la aplicación por sí misma, su organización en componentes, y su despliegue en nodos de ejecución. Estas vistas proporcionan una oportunidad de establecer correspondencias entra las clases y los componentes e implementación y nodos.

Hay dos vistas físicas: la vista de implementación y la vista de despliegue.

- *Vista de implementación*: La vista de implementación modela los componentes de un sistema, a partir de los cuales se construye la aplicación, y modela las dependencias entre los componentes, para poder determinar el impacto de un cambio propuesto. La vista de implementación se representa en diagramas de componentes.

- *Vista de despliegue*: La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos. Un nodo es un recurso de ejecución, tal como una computadora, un dispositivo, o memoria. Esta vista permite determinar las consecuencias de la distribución y de la asignación de recursos. La vista de despliegue se representa en diagramas de despliegue.

Diagramas: Los diagramas son las gráficas que describen el contenido de una vista. UML tiene nueve tipos de diagramas que son utilizados en combinación para proveer todas las vistas de un sistema: diagramas de caso de uso, de clases, de objetos, de estados, de secuencia, de colaboración, de actividad, de componentes y de despliegue.

Símbolos o Elementos de modelo: Los conceptos utilizados en los diagramas son los elementos de modelo que representan conceptos comunes orientados a objetos, tales como clases, objetos y mensajes, y las relaciones entre estos conceptos incluyendo la asociación, dependencia y generalización. Un elemento de modelo es utilizado en varios diagramas diferentes, pero siempre tiene el mismo significado y simbología.

Reglas o Mecanismos generales: Proveen comentarios extras, información o semántica acerca del elemento de modelo; además proveen mecanismos de extensión para adaptar o extender UML a un método o proceso específico, organización o usuario.

II.3. Aplicación en los procesos de negocio

La evolución de UML ha hecho converger en una misma notación la especificación de componentes de software (desde su análisis hasta su implementación), y el modelado de procesos de negocio de una organización (mapas de procesos). Esta unificación tiene un impacto altamente positivo en los entornos empresariales donde conviven ingenieros de procesos y desarrolladores de software. La ausencia de un vocabulario común y, la incompatibilidad de los modelos de especificación, ha impedido hasta ahora poder establecer un espacio de colaboración, en el cual, todos los agentes involucrados puedan definir y resolver de una manera eficiente los problemas de información y comunicación de los usuarios finales.

A través de la notación UML se puede comunicar y compartir el conocimiento de una arquitectura (conceptual, de diseño, y de implementación), gracias a la combinación simultánea de distintas perspectivas:[8]

1. Definir conceptos a través de sus atributos distintivos y trazar sus límites.
2. Formalizar unas reglas de relación y actuación.
3. Hacer visible la granularidad y entrelazamiento de unos elementos.
4. Mantener la trazabilidad entre la concepción de un problema y su solución.
5. Tomar decisiones y evaluar su impacto en la arquitectura de manera ágil.
6. Organizar la experiencia de los usuarios en patrones de conocimiento.
7. Comprobar la coherencia, el uso y que el modelo final sea completo.
8. Alinear la solución tecnológica con la cadena de valor de los Actores.
9. Usar un vocabulario controlado dentro de un espacio de colaboración.

UML entrega una forma de modelar aspectos conceptuales como lo son procesos de negocio y funciones de sistema, además de aspectos concretos como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables. Mientras que ha habido muchas notaciones y métodos usados

[8] http://www.bit-net.org/tribuna/tribuna20/interactiva.htm

para el diseño orientado a objetos, ahora los modeladores sólo tienen que aprender una única notación.

Un documento separado dentro de la especificación UML define clases y estereotipos de asociación específicos que extienden UML hasta cubrir conceptos de modelado de negocio. Esto incluye una clase como un actor, un trabajador, o una entidad y típicamente una comunicación simple, o una suscripción entre un origen y un objetivo.

Modelado de negocios con UML provee una metodología para modelar procesos de negocio basada en el uso del mismo lenguaje (UML) que utilizan los equipos de software para modelar sus sistemas. Utilizando la misma notación en todos los dominios (negocio, sistemas y usuarios) de la organización, se logra incrementar considerablemente la calidad de la comunicación entre analistas de negocio y los equipos de software, además es más probable la inclusión de consideraciones de negocio en los requerimientos de sistema, eventualmente facilitando la construcción de mejores sistemas.

ANÁLISIS DEL PROCESO DE VENTAS CON UML

III. ANÁLISIS DEL PROCESO DE VENTAS CON UML

III.1. Análisis del proceso de ventas

Hay dos razones del porqué los procesos son la clave para un negocio. La primera razón es la eficiencia, ya que una vez que se establece un proceso y en él se muestran los pasos a seguir para realizar una tarea y éstos resultan efectivos, no tiene ningún sentido buscar nuevos pasos cada vez que se realiza esa tarea, de hacerlo así el resultado será una pérdida de tiempo. Por ello, tener documentación que se use como herramienta ahorra tiempo, energía y recursos. La segunda razón es la escalabilidad. Para que un equipo de trabajo se desarrolle, debe ser posible delegar actividades y tareas. Si se tiene un proceso, se puede entrenar gente para que lo ejecute. Con la capacitación adecuada y herramientas analíticas, el personal puede reunir los elementos necesarios para llevar a cabo una actividad, administrar el trabajo de otros, o concebir planes estratégicos.

Todo negocio opera como una colección de procesos relacionados. Cada proceso inicia con una especie de solicitud y termina con la entrega de un servicio o producto. Algunos procesos sirven a clientes externos o usuarios, mientras que otros son puramente internos o de naturaleza administrativa.

Es necesario hacer un análisis profundo para entender perfectamente los procesos de negocio del cliente, el cual se puede hacer entrevistándolo directamente, o a una persona bien informada designada por éste, y pidiendo a la persona entrevistada examinar los procesos relevantes gradualmente, de esta manera se podrá saber lo necesario e indispensable y en base a eso, determinar lo que es necesario hacer. El análisis de requerimientos es el primer paso del análisis del sistema.

El primer paso para la creación del diseño de un proceso es identificar los objetos empresariales, es decir los componentes que proporcionarán la funcionalidad necesaria. A continuación, se debe identificar los comportamientos, atributos y relaciones de cada objeto.

Al analizar un escenario de uso, lo primero que hay que hacer es identificar los objetos presentes en dicho escenario. Un escenario muestra una secuencia diferente de interacciones entre actores y el proceso. Un actor es un agente externo al sistema pero que interactúa con él. En el siguiente capítulo se encuentra una definición más detallada de actor, así como su representación, tipos, responsabilidades y su uso en los diagramas.

Un objeto es, por lo general, una entidad o proceso empresarial que aparece en el escenario de uso. Es un ente tangible que pertenece a una clase, siendo esto su principal característica. Un objeto pueden ser hechos, por ejemplo, una clase llamada VENTA que tiene objetos como: No. de venta, fecha, vendedor, etc.

Una clase representa un concepto del negocio describible y plenamente identificable. Es un conjunto de variables, llamados atributos, y funciones, llamadas métodos, que trabajan sobre esas variables. Describe un conjunto de objetos con características y comportamiento idéntico.

En el Capítulo IV se encuentra una mayor información de clase, así como su uso en los diagramas de clase.

Continuando con el diseño de un proceso, una vez que se hayan identificado los objetos que constituyen la base del escenario, es necesario disponer de objetos adicionales para que el escenario funcione, aunque dichos objetos no aparecen de forma específica en el escenario, estos se pueden identificar examinando los comportamientos sin objetos aparentemente asociados. Para identificar estos objetos, es necesario, en primer lugar, identificar los comportamientos correspondientes, también denominados *métodos o servicios*.

Un método es la implementación de una operación, mediante el cual se especifica el algoritmo o procedimiento que da lugar a los resultados de una operación.[9]

Para identificar los comportamientos de los objetos, es necesario evaluar, en primer lugar, las acciones que tienen lugar en el escenario, es decir las acciones que el usuario podrá realizar.

A continuación, se debe seguir evaluando cada una de las acciones del escenario hasta identificar todos los objetos necesarios y sus comportamientos asociados.

Una vez que se han identificado los comportamientos, el siguiente paso consiste en identificar los atributos, también denominados *propiedades*, de los objetos definidos. Los atributos son los elementos de los que la aplicación necesita realizar un seguimiento. Se trata de marcadores de posición en los que los datos se retienen o almacenan.

Los atributos se obtienen del análisis de los comportamientos del escenario y de la extracción de los elementos que se deben almacenar o controlar.

Una vez que se han definido los objetos, sus comportamientos y atributos, el siguiente paso es identificar sus relaciones, que son asociaciones lógicas entre objetos.

Para identificarlas, es necesario analizar la interacción entre los distintos objetos, y de esta manera ver que objetos están relacionados unos con otros. Por lo ejemplo, el objeto X, con cierto comportamiento, tiene una relación con el objeto Y, ya que contiene varios de estos objetos.

Una relación representa una conexión semántica materializada entre elementos de un modelo. Las relaciones entre clasificadores son asociación, generalización, flujo, y varias clases de dependencia.

Un clasificador es un elemento del modelo que describe características estructurales y de comportamiento. Las clases, actores, componentes, tipos de datos, interfaces, nodos, señales, subsistemas y casos de uso son tipos de clasificadores. El tipo más general de clasificador es la clase. Otros tipos pueden tener un cierto parecido intuitivo con las clases, con ciertas restricciones sobre el contenido y la utilización, aunque cada tipo de clasificador está representado por su propia clase en el metamodelo. En general la mayoría de las propiedades de las clases se aplican a los clasificadores, con ciertas restricciones según el tipo de clasificador.

Un concepto importante relacionado con las relaciones es el de asociación. Una asociación en general es una línea que une dos o más símbolos. Pueden tener varios tipos de símbolos, que definen su semántica y características.

[9] El lenguaje unificado de modelado. Manual de referencia. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 2000.

Las asociaciones llevan la información sobre relaciones entre objetos en un sistema. Cuando se ejecuta un sistema, los enlaces entre objetos se crean y se destruyen. Las asociaciones son las que mantienen unido un sistema.[10]

Los tipos de asociaciones entre clases presentes en un diagrama estático son:

1. Asociación binaria
2. Asociación n-aria
3. Composición
4. Generalización
5. Refinamiento

Cada asociación puede presentar algunos elementos adicionales que dan detalle a la relación, como son:

Rol: Identificado como un nombre al final de la línea, describe la semántica de la relación en el sentido indicado.

Multiplicidad: Describe la cardinalidad de la relación. A la relación numérica asociada se le denomina cardinalidad, en UML se usan rangos para determinar los valores que pueden asumir una cardinalidad dada. Esta se representa como: Mínimo..Máximo. El símbolo * (asterisco) representa un número N>0 (cero), cuyo valor exacto no está determinado.

La especificación de la multiplicidad puede ser de cualquiera de las siguientes formas:

- Uno y sólo uno
- 0..1 Cero o uno
- M..N Desde M hasta N (enteros naturales)
- 0..* Cero o muchos
- 1..* Uno o muchos (al menos uno)

Una asociación binaria se identifica como una línea sólida que une dos clases. Representa una relación de algún tipo entre las dos clases, no muy fuerte (es decir, no se exige dependencia existencial ni encapsulamiento).

Un ejemplo de este tipo de asociación es la que se muestra en la figura III.1, la cual muestra la relación entre una compañía y sus empleados. En este caso la relación recibe el nombre genérico *Trabaja Para*, la compañía tiene uno o más instancias de la clase *Persona* denominadas *empleado* y cada empleado conoce su *empleador* (en este caso único).

[10] http://www.cs.ualberta.ca/~pfiguero/soo/uml/estr_estatica01.html



Figura III.1. Representación de una asociación binaria

Una instancia es una entidad de ejecución con identidad, es decir, algo que puede ser distinguido de otras entidades de ejecución. Tiene un valor en todo momento. A lo largo del tiempo, el valor puede cambiar en respuesta a operaciones sobre él.

Los elementos y diagramas para comportamiento de UML describen las secuencias válidas de las instantáneas que pueden ocurrir como resultado de efectos del comportamiento externo e interno.

La llamada relación de agregación, también conocida como composición. Este tipo de relación describe relaciones de todo-partes, es decir proporciona una idea general de un todo y de sus partes que lo componen. Describe como está conformado cada objeto de una clase. Este tipo de relación es utilizada cuando hay información repetitiva que forma parte de un concepto mayor.

Esta relación puede ser caracterizada con precisión determinando las relaciones de comportamiento y estructura que existen entre el objeto agregado y cada uno de sus objetos componentes.

Una agregación se podría caracterizar según la manera de cómo se responda a las siguientes preguntas:

1. ¿Puede el objeto parte comunicarse directamente con objetos externos al objeto agregado?

   No => inclusiva
   Si => no inclusiva

2. ¿Puede cambiar a composición del objeto agregado?
   Si => dinámica
   No => estática

Una composición es una asociación fuerte, que implica tres cosas

- Dependencia existencial. Esto implica que el elemento dependiente desaparece al destruirse el que lo contiene y, si es de cardinalidad 1, es creado al mismo tiempo.

- Hay una pertenencia fuerte, por lo que se puede decir que el objeto contenido es parte constitutiva y vital del que lo contiene.

- Los objetos contenidos no son compartidos, esto es, no hacen parte del estado de otro objeto.

Una composición se denota dibujando un rombo relleno del lado de la clase que contiene a la otra en la relación.

Existe también una relación de composición menos fuerte (no se exige dependencia existencial, por ejemplo) que es denotada por un rombo sin rellenar en uno de los extremos.

En la figura III.2 se muestra una representación de la relación de agregación en la que la cama representa el todo, y las patas y la cabecera son las partes. La relación de la cama con las patas es dependiente por lo que el rombo es relleno y la relación de la cama con la cabecera no exige dependencia, por lo que el rombo no está relleno.

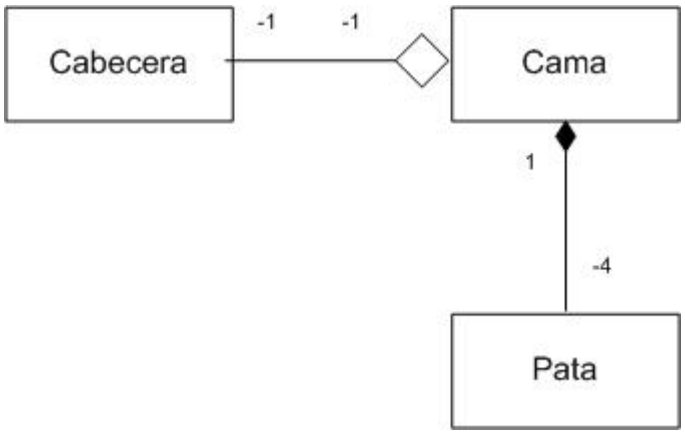


Figura III.2. Representación de la relación de agregación.

Es importante resaltar que existe otro tipo de relación, la relación de generalización, llamada también *herencia*, que se da sólo cuando un objeto es definido por otro. Esta relación se da cuando existen conceptos generales que agrupan conceptos particulares, con quienes comparten similitudes pero de los cuales son diferentes. Se representa dibujando un triángulo sin rellenar en el lado de la superclase.

La generalización permite gestionar la complejidad mediante un ordenamiento taxonómico de clases, se obtiene usando los mecanismos de abstracción de generalización y/o especialización. Las subclases heredan propiedades de sus clases padre, es decir, atributos y operaciones (y asociaciones) de la clase padre están disponibles en sus clases hijas. La generalización y especialización son equivalentes en cuanto al resultado: la jerarquía y herencia establecidas.

En la figura III.3 se muestra un ejemplo de cómo se representa la relación de generalización. En este ejemplo se puede ver que los productos de seguridad así como los productos que son sistemas operativos, son software y por lo tanto heredan las propiedades de software, el cual a su vez es una subclase de la clase producto por lo que hereda las propiedades de éste, como podría ser un precio, un número de serie, etc. Estas mismas propiedades son heredades por la subclase hardware quien a su vez hereda todas sus propiedades a los productos de comunicaciones.

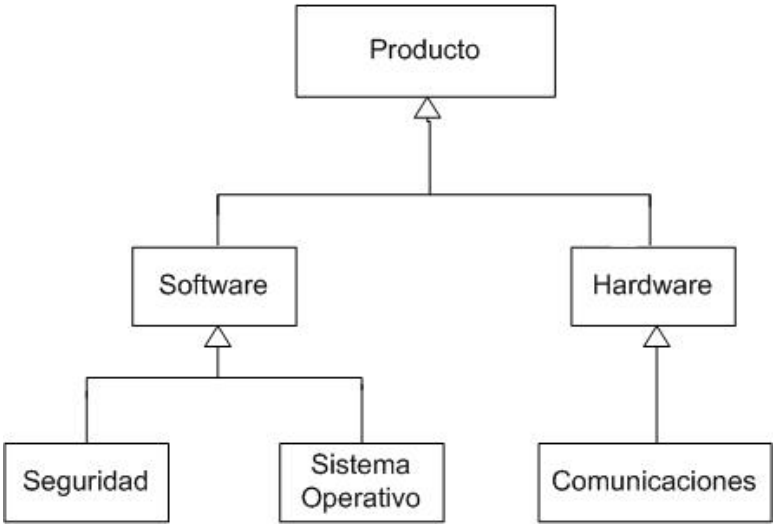


Figura III.3. Relación de generalización

Una jerarquía de *generalización-especialización* clasifica un concepto en subconceptos con la finalidad de proveer un mecanismo para el manejo de semejanzas y diferencias.

La relación de refinamiento es una relación entre dos elementos donde uno de ellos especifica de forma completa al otro que ya ha sido especificado con cierto detalle. Los refinamientos se indican mediante una flecha de dependencia (una flecha discontinua con la punta en el elemento más general y el origen en el elemento más específico) que posee la palabra reservada «refine».

Como se ha venido mencionando, UML provee un sistema de conceptos y de relaciones para modelar sistemas como gráficos de elementos de modelado. Sin embargo algunas cosas se expresan mejor lingüísticamente, usando un lenguaje textual. Una restricción es una expresión booleana representada como una cadena interpretable en un determinado lenguaje. Para expresar restricciones se puede utilizar el lenguaje natural, notación de teoría de conjuntos, lenguajes de restricciones o varios lenguajes de programación.[11]

En base a lo mencionado anteriormente se puede resumir que a grandes rasgos las tareas a realizar son las siguientes:

1. Identificar los objetos empresariales del escenario.
2. Identificar los comportamientos de estos objetos.
3. Identificar sus atributos o propiedades.
4. Identificar las relaciones lógicas entre ellos.

Una vez que se hayan realizado, completado y documentado estas tareas para cada escenario de uso, finaliza la fase de análisis de diseño del proceso.

[11] Modelado de objetos con UML. P. Muller. Ed. Gestión. 2000.

III.2. Definición de requerimientos

Como se mencionó anteriormente, la definición de requerimientos es el primer paso del análisis del sistema, en este proceso el analista se reúne con el cliente y/o usuario (un representante institucional, departamental o cliente particular), para identificar las metas globales, analizar las perspectivas del cliente, sus necesidades y requerimientos, sobre la planificación temporal y presupuestal, líneas de mercado y otros puntos que puedan ayudar a la identificación y desarrollo del proyecto.

Algunos autores suelen llamar a esta parte *análisis de requisitos* y lo dividen en cinco partes[12]:

- Reconocimiento del problema.
- Evaluación y Síntesis.
- Modelado.
- Especificación.
- Revisión.

Antes que esta reunión suceda, el cliente puede preparar un documento conceptual del proyecto, en el que se plasmen los requerimientos por parte de la empresa, aunque es recomendable que este se elabore durante la reunión, para que de esta manera quede plasmados una vez que ha sido analizado y revisado por ambas partes, ya que lo más probable es que el documento hecho por el cliente tenga modificaciones después de realizar el análisis de las necesidades de la empresa.

Las modificaciones que pueda tener el documento hecho por el cliente son debido a que muchas veces este no tiene claro cual es la problemática, o no conoce cómo están definidos y entrelazados los procesos (si es que se tienen). Otro aspecto que hay que tomar en cuenta son los requerimientos y el tiempo para su realización, lo cual también puede ser la causa de la modificación del documento, ya que lo que muchas ocasiones ocurre es que cuando se emprende el desarrollo de un proyecto de sistemas, estos no son realistas para su realización sin tener perdidas económicas y frustración profesional.

Para la definición de requerimientos es importante tomar en cuenta la viabilidad y el análisis de riesgos, los cuales están relacionados de muchas maneras, si el riesgo del proyecto es alto, la viabilidad de producir software de calidad se reduce, sin embargo se deben tomar en cuenta tres áreas principales de interés: la viabilidad económica, la viabilidad técnica, la viabilidad legal.

Para verificar la viabilidad económica es preciso realizar una evaluación de los costos de desarrollo, comparados con los ingresos netos o beneficios obtenidos del producto o sistema desarrollado.

El análisis de la viabilidad técnica puede hacerse por medio de un estudio de funciones, rendimiento y restricciones que puedan afectar la realización de un sistema aceptable.

[12] Ingeniería del Software. Un enfoque práctico. R. Pressman. Ed. McGraw-Hill. 1998.

Por último, para la viabilidad legal es preciso determinar cualquier posibilidad de infracción, violación o responsabilidad legal en que se podría incurrir al desarrollar el Sistema.

UML tiene casos de uso (use-cases) para capturar los requerimientos del cliente. A través del modelado de estos, los actores externos que tienen interés en el sistema son modelados con la funcionalidad que ellos requieren del sistema. Los actores y los casos de uso son modelados con relaciones y tienen asociaciones entre ellos o éstas son divididas en jerarquías. Los actores y casos de uso son descritos en un diagrama use-case. Cada uno de estos diagramas es descrito en texto y especifica los requerimientos del cliente: lo que él (o ella) espera del sistema sin considerar la funcionalidad que se implementará. Un análisis de requerimientos puede ser realizado también para procesos de negocios, como será en este caso, no solamente para sistemas de software.

III.3. Asignación de responsabilidades

En el Capítulo I se definió lo que es un proceso, es decir, la manera en que se hace el conjunto de actividades o tareas en una organización, es por eso que en todo proceso es necesario:

- Que sea identificada cada tarea.
- Que se especifique el tiempo en el cual debe ser completada cada tarea.
- Que se indiquen claramente las relaciones e interdependencia entre las diferentes tareas.
- Que se designen personas o equipos específicos para concluir cada tarea.

Cumpliendo los anteriores puntos se tendrá un proceso efectivo y eficiente.

Como se muestra en los puntos anteriores, una vez que la tarea es identificada, y que los tiempos son definidos al igual que las relaciones con otras tareas, es importante asignar responsables para la realización de esa tarea. La asignación de responsabilidades nos ayuda a saber quién hace qué.

Hay un refrán que dice: *Todos están de acuerdo que alguna cosa necesita hacerse, cualquiera dice alguno deberá hacerla y ninguno la hizo.* La importancia de la asignación de responsabilidades es el asegurar que lo mencionado en el refrán no suceda, y esto se logra asignando responsables que cubran todas las tareas que son identificadas[13].

La asignación de responsabilidades debe hacerse debido a que, en todo proceso es importante hacer un monitoreo efectivo que muestre el desempeño de éstos, sin una correcta asignación de responsabilidades, pueden fallar los mecanismos de responsabilidad y rendición de informes. Otro punto importante por lo que la asignación de actividades debe ser muy detallada y minuciosa, es porque a menudo se olvidan las tareas aparentemente irrelevantes pero críticas, lo cual no pasará si los responsables están claramente identificados y saben que de ellos depende concluir debidamente esa tarea. Para esto puede hacerse una lista de

[13] http://www.fao.org/documents/show_cdr.asp?url_file=/DOCREP/005/X0475S/x0475s0s.htm

responsabilidades de todas las actividades necesarias para la producción de cualquier resultado. Éstas deben incluir todos los pasos necesarios.

Es necesario que las personas a quienes se asigne una responsabilidad estén plenamente conscientes de ella y de cómo esta se relaciona con otras actividades. Para esto es necesario informar y capacitar a cada responsable de las relaciones y actividades realizadas en conjunto con otras áreas o responsables. Hay que tener presente que nadie puede hacer su trabajo adecuadamente si no tiene en claro cuál es ese trabajo. Existe un aspecto personal y positivo en la asignación de responsabilidades, y esto es, que cada responsable tenga plena conciencia de las responsabilidades que le corresponden y que debe cumplir con ellas, ya que para cualquier persona, el tener en mente que es parte de una actividad incrementa su autoestima y permite mejorar su desempeño.

En la figura III.4 se muestra una tabla en la que se lleva el control de actores y responsabilidades dentro de una empresa, de esta manera queda definido quién realiza qué, es decir los actores y sus responsabilidades.

Actor	Responsabilidades
Vendedor	Asesorar al cliente
	Dar seguimiento a proyectos
	Realizar ventas y afectar el inventario
Cuentas por cobrar	Dar seguimiento a ventas realizadas con crédito
Marketing	Organizar eventos de marketing
	Planear promociones
Tráfico de mercancías	Realizar órdenes de tráfico
	Planear la entrega de mercancía
	Verificar recepción de mercancías

Figura III.4. Tabla de actores y responsabilidades

Con la asignación de responsabilidades, se delimita un sistema, lo cual es crucial, debido a que:

- Se delimita el alcance del sistema, indicando que actores participan.
- Se identifican responsabilidades mayores de un actor para delimitar el alcance del sistema en términos que el actor comprenda
- Se modelan los actores en un diagrama llamado diagrama de contexto

El diagrama de contexto muestra la relación existente entre la organización y su entorno (sectores públicos, privados, organismos internacionales, entre otros), así como los requerimientos de información (proporcionada por las fuentes) y la información producida por ésta (empleada por los usuarios).

En la siguiente figura III.5 se muestra el diagrama de contexto para una empresa dedicada a la venta de productos y servicios.

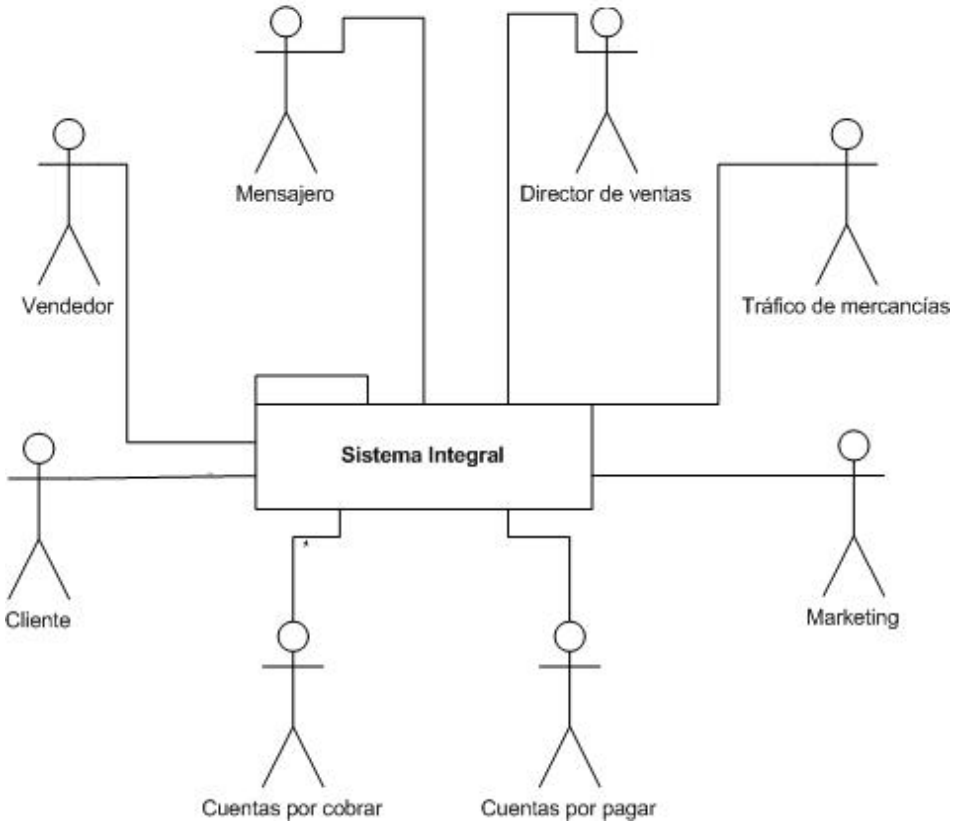


Figura III.5. Representación de un diagrama de contexto

Una vez que se ha establecido “quién” hace “qué”, debe definirse el “cuándo”, esto se hace a través de los planes. Estos deben hacerse con anticipación para adquirir materiales para el lugar de trabajo (en caso de ser necesario), a tiempo, y que no cause demoras. El concepto de planeamiento es simple, es un plan en el que se plasma la forma en que una persona cree que se puede alcanzar un objetivo.

MODELADO DEL PROCESO DE VENTAS CON DIAGRAMAS EN UML

IV. MODELADO DEL PROCESO DE VENTAS CON DIAGRAMAS EN UML

Un diagrama es una representación gráfica de una colección de elementos del modelo, construida a menudo como un gráfico conexo de arcos (relaciones) y de vértices (otros elementos modelo). UML ofrece diagramas de casos de uso, diagramas de secuencia, diagramas de colaboración, diagramas de objetos, diagramas de estados, diagramas de actividad, diagramas de clases, diagramas de componentes y diagramas de despliegue.

IV.1. Modelado de diagramas de Caso de Uso

IV.1.1. Descripción

Los casos de uso no son parte del diseño (cómo), sino parte del análisis (qué). Es una técnica para capturar información de cómo un sistema o negocio trabaja o de cómo se desea que trabaje, de forma que al ser parte del análisis nos ayudan a describir qué es lo que el sistema debe hacer. Los casos de uso son lo que hace el sistema desde el punto de vista del usuario. Es decir, describen un uso del sistema y cómo este interactúa con el usuario. Estrictamente no pertenece al enfoque orientado a objeto, ya que es solo una técnica para captura de requisitos.

En el diagrama de casos de uso se representa también el sistema como una caja rectangular con el nombre en su interior. Los casos de uso están en el interior de la caja del sistema y los actores fuera, cada actor está unido a los casos de uso en los que participa mediante una línea.

Los elementos que pueden aparecer en un diagrama de casos de uso son: actores, casos de uso y relaciones entre casos de uso.

Un caso de uso es una descripción de la secuencia de interacciones que se producen entre un actor y el sistema, cuando el actor usa el sistema para llevar a cabo una tarea específica. Cada caso de uso es una operación completa desarrollada por los actores y por el sistema en un diálogo. El conjunto de casos de uso representa la totalidad de operaciones desarrolladas por el sistema.[14] Se representan en un diagrama por una elipse, lo cual denota un requerimiento solucionado por el sistema. Van acompañados de un nombre significativo dentro de la elipse, como se muestra en la figura IV.1.



Figura IV.1. Representación de un caso de uso

[14] El lenguaje unificado de modelado. Manual de referencia. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 2000.

Los casos de uso se determinan observando y precisando, actor por actor, las secuencias de interacción, los escenarios, desde el punto de vista del usuario. Los casos de uso intervienen durante todo el ciclo de vida. El proceso de desarrollo estará dirigido por los casos de uso. Un escenario es una instancia de un caso de uso.

A continuación se muestran algunas características de los casos de uso:

- Permiten definir los límites del sistema y las relaciones entre el sistema y el entorno.
- Son descripciones de la funcionalidad del sistema independientes de la implementación.
- Los casos de uso cubren la carencia existente en métodos previos (OMT, Booch) en cuanto a la determinación de requisitos.
- Los casos de uso separan el conjunto de necesidades atendiendo a la categoría de usuarios que participan en el mismo.
- Están basados en el lenguaje natural, es decir, es accesible por los usuarios.

Debido a que los actores juegan un importante papel en la representación de los diagramas de caso de uso, es importante tener definido que es un actor. Un actor algo con comportamiento, como una persona (identificada por un rol), un sistema informatizado u organización y que realiza algún tipo de interacción con el sistema, que necesita o usa algunos de los casos de uso. Se representa mediante una figura humana dibujada con palotes, como se muestra en la figura IV.2 y es acompañado de un nombre significativo, si es necesario. Esta representación sirve tanto para actores que son personas como para otro tipo de actores.



Figura IV.2. Representación de un actor en los diagramas de casos de uso

Una misma persona física puede interpretar varios papeles como actores distintos, el nombre del actor describe el papel desempeñado.

Los actores pueden ser:

- Principales: personas que usan el sistema.
- Secundarios: personas que mantienen o administran el sistema.
- Material externo: dispositivos materiales imprescindibles que forman parte del ámbito de la aplicación y deben ser utilizados.
- Otros sistemas: sistemas con los que el sistema interactúa.

Es posible obtener a los actores de un diagrama de casos de uso a través de las siguientes preguntas:

- ¿Quién utilizará la funcionalidad principal del sistema (actores primarios)?
- ¿Quién necesitará soporte del sistema para realizar sus actividades diarias?
- ¿Quién necesitará mantener, administrar y trabajar el sistema (actores secundarios)?
- ¿Qué dispositivos de hardware necesitará manejar el sistema?
- ¿Con qué otros sistemas necesitará interactuar el sistema a desarrollar?
- ¿Quién o qué tiene interés en los resultados (los valores) que el sistema producirá?

En la figura IV.3, se puede observar como se representa un diagrama de caso de uso y los elementos que lo conforman.

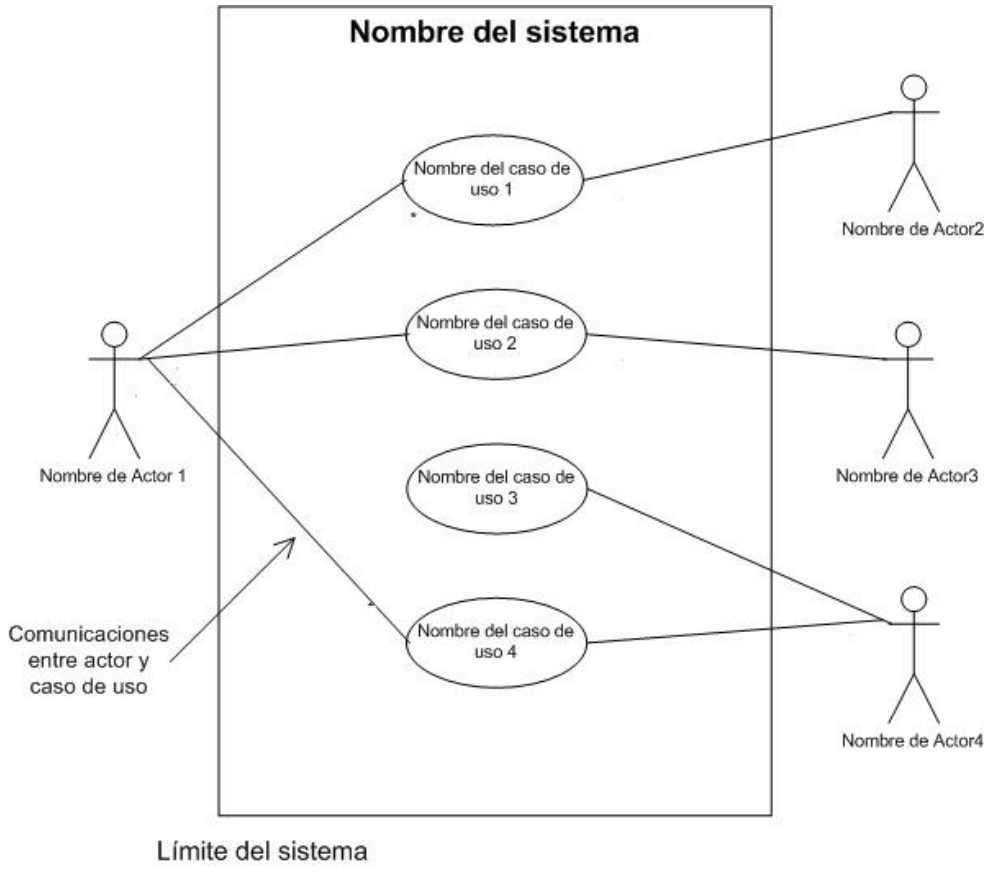


Figura IV.3. Representación de un diagrama de caso de uso

En UML se presentan tres tipos de relación para los diagramas de uso, que son representadas por líneas dirigidas entre ellos (del elemento dependiente al independiente), estas relaciones son[15]:

- Comunicación («communicates»): Relación entre un actor y un caso de uso, denota la participación del actor en el caso de uso determinado.

- Uso («uses»). Relación entre dos casos de uso, denota la inclusión del comportamiento de un escenario en otro. Un caso de uso base incorpora explícitamente a otro caso de uso en un lugar especificado en dicho caso base. Se utiliza para modelar un caso de uso que utiliza (o 'usa') TODA la funcionalidad de otro caso de uso. Este 'uso' se implementa mediante algún mecanismo de invocación. En este contexto, el caso de uso que es invocado tiene dos características principales: no comparte el diseño de interfaz del caso de uso que invoca y resuelve un solo problema (no es una operación). En la figura IV.4 se muestra una representación gráfica de este tipo de relación.

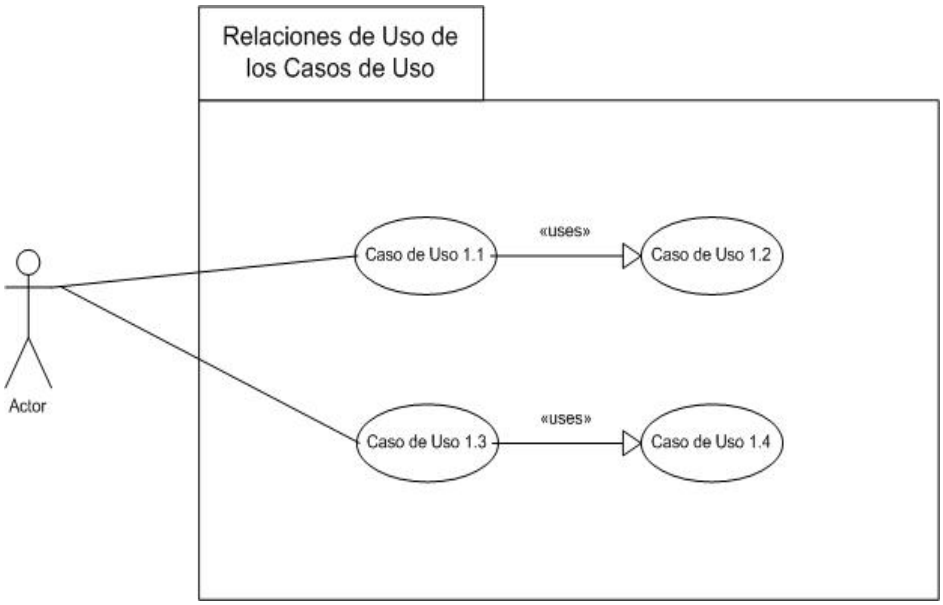


Figura IV.4. Representación de relación «uses»

- Extiende («extends»): Relación entre dos casos de uso, denota cuando un caso de uso es una especialización de otro. El caso de uso origen extiende el comportamiento del caso de uso destino. Es utilizado cuando un caso de uso base tiene ciertos puntos (puntos de extensión) en los cuales, dependiendo de ciertos criterios, se va a realizar una interacción adicional. El caso de uso que extiende describe un comportamiento opcional del sistema (a diferencia de la relación uso que se da siempre que se realiza la interacción descrita). Este tipo de relaciones son utilizadas en escenarios en los que se necesita: Modelar implementaciones de políticas de la empresa, representar comportamientos adicionales de un caso de uso, modelar comportamientos de casos de uso en situaciones complejas, modelar situaciones de excepción

[15] El lenguaje unificado de modelado. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 1999.

(errores). En la figura IV.5 se muestra una representación gráfica de este tipo de relación.

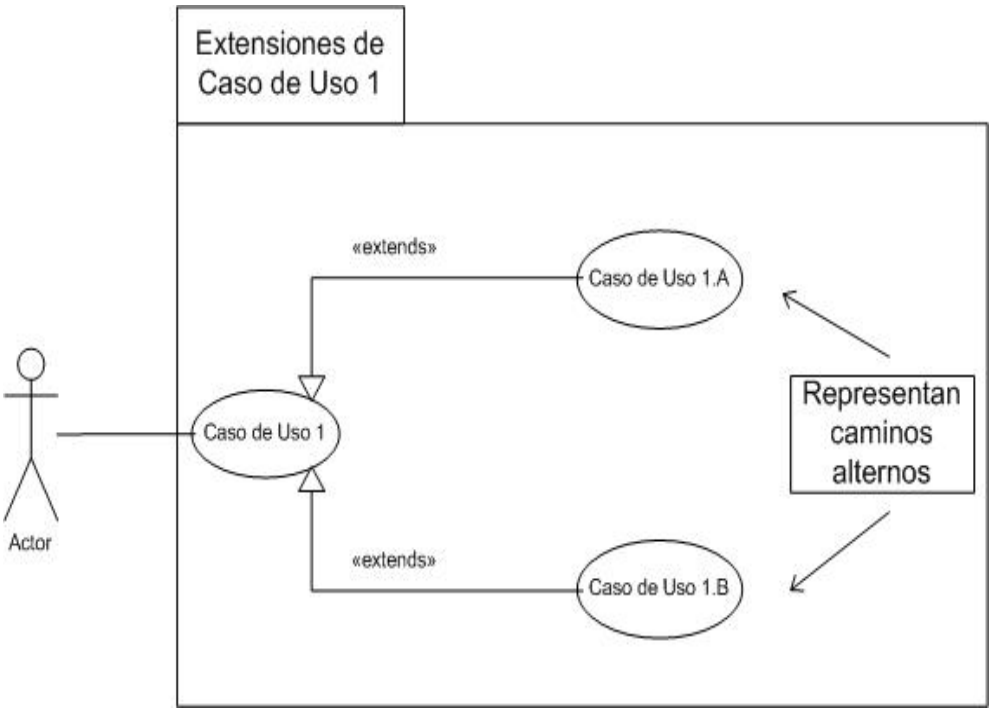


Figura IV.5. Representación de relación «extends»

Los tipos de relación mencionados se representan como una dependencia etiquetada con el estereotipo correspondiente (« o »), de tal forma que la flecha indique el sentido en el que debe leerse la etiqueta. Junto a la etiqueta «» se puede detallar el/los puntos de extensión del caso de uso base en los que se aplica la extensión.

Para construir un caso de uso hay que tener en cuenta que este debe ser simple, inteligible, claro y conciso. El proceso para encontrar casos de uso inicia encontrando al actor o actores previamente definidos. Por cada actor identificado, hay que realizar las siguientes preguntas:

- ¿Qué funciones del sistema requiere el actor? ¿Qué necesita hacer el actor?
- ¿El actor necesita leer, crear, destruir, modificar o almacenar algún tipo de información en el sistema?
- ¿El actor debe ser notificado de eventos en el sistema o viceversa? ¿Qué representan esos eventos en términos de funcionalidad?
- ¿El trabajo diario del actor podría ser simplificado o hecho más eficientemente a través de nuevas funciones en el sistema? (Comúnmente, acciones actuales del actor que no estén automatizadas).

Otras preguntas que nos ayudan a encontrar casos de uso pero que no involucran actores son:

- ¿Qué entradas/salidas necesita el sistema? ¿De dónde vienen esas entradas o hacia dónde van las salidas?

- ¿Cuáles son los mayores problemas de la implementación actual del sistema?

La descripción del caso de uso comprende:

1. El inicio: ¿cuándo y qué actor lo produce?
2. El fin: ¿cuándo se produce y qué valor devuelve?
3. La interacción actor-caso de uso: ¿qué mensajes intercambian ambos?
4. Objetivo del caso de uso: ¿qué lleva a cabo o intenta?
5. Cronología y origen de las interacciones
6. Repeticiones de comportamiento: ¿qué operaciones son iteradas?
7. Situaciones opcionales: ¿qué ejecuciones alternativas se presentan en el caso de uso?

Para ilustrar los conceptos y diagramas de UML, nos basaremos en un ejemplo simple, con el propósito de destacar varias construcciones de UML de manera sencilla. El ejemplo es una taquilla de cine que ha automatizado sus operaciones.

En la figura IV.6 se muestra un diagrama de casos de uso para el ejemplo de la taquilla. Los actores son el vendedor, el supervisor, y el cliente. Los casos de uso incluyen el comprar entradas, a través del cliente o del vendedor, comprar suscripciones (solamente a través del vendedor), y examinar las ventas totales a manera de reportes (a petición del supervisor).

El comprar entradas o suscripciones pueden hacerse por medio de una tarjeta de crédito, por lo cual, en el diagrama se muestra que incluyen un fragmento común estos dos casos de uso, que es hacer cargos a la tarjeta de crédito.

Como se muestra en la figura IV.6, la compra de boletos para la entrada al cine puede hacerla el cliente directamente o el vendedor, para ambos casos se necesita obligatoriamente el cobro de los boletos, es por ello que se utiliza la relación «uses», por lo que el caso de uso *comprar entradas* a través de una invocación usa toda la funcionalidad del otro caso de uso *cobrar*, con esto se denota la inclusión de su comportamiento. Para el caso de compra de suscripciones, solamente se puede hacer por medio del vendedor, el cual también el caso de uso *comprar suscripción* utiliza la relación «uses» con el caso de uso *cobrar.* A su vez, el caso de uso *cobrar,* puede hacer uso del servicio de tarjeta de crédito por lo cual se encuentra relacionado con éste. Por otra parte el caso de uso *examinar ventas* puede ser revisado a través de reportes por parte del supervisor.

Como se muestra, a grandes rasgos este es el diagrama de casos de uso para una taquilla, en el cual se describen los actores, casos de uso y relaciones entre ellos, de esta manera es mucho más fácil entender como se está trabajando desde el punto de vista del usuario.

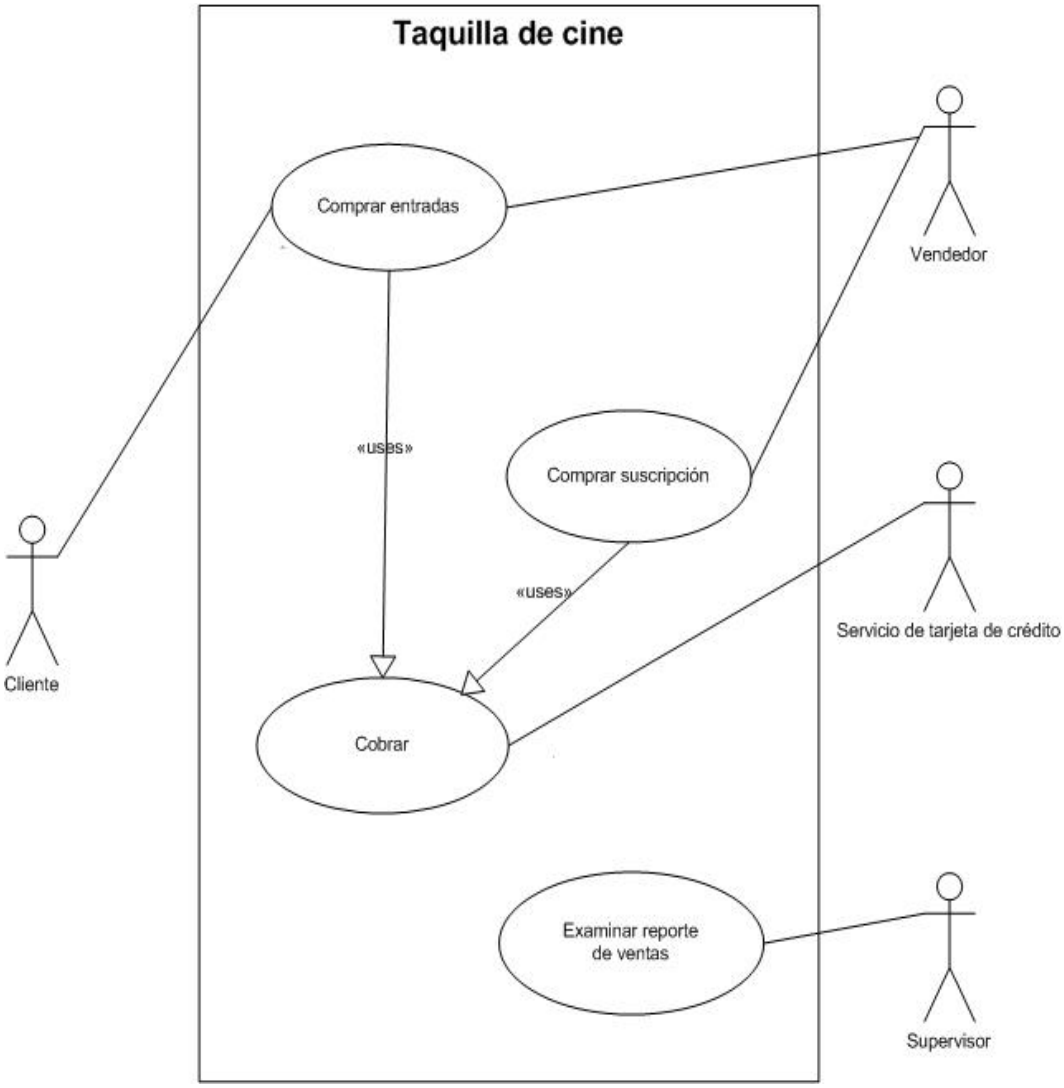


Figura IV.6. Ejemplo práctico de un diagrama de caso de uso

IV.2. Diagramas de Iteración

La vista de interacción describe secuencias de intercambios de mensajes entre los roles que implementan el comportamiento de un sistema. Un rol clasificador o simplemente "un rol", es la descripción de un objeto, que desempeña un determinado papel dentro de una interacción distinto de los otros objetos de la misma clase. Esta visión proporciona una vista integral del comportamiento del sistema, es decir, muestra el flujo de control a través de muchos objetos.

Una interacción es un conjunto de mensajes dentro de una colaboración que son intercambiados por roles de clasificador a través de roles de asociación. Cuando una colaboración existe en tiempo de ejecución, los objetos ligados a roles de clasificador intercambian instancias de mensajes a través de los enlaces ligados a los roles de asociación. Una interacción modela la ejecución de una operación, caso de uso u otra entidad de comportamiento.

Un mensaje es una comunicación unidireccional entre dos objetos, un flujo de objetos con la información de un remitente a un receptor. Un mensaje puede tener parámetros que transporten valores entre los objetos. Un mensaje puede ser una señal (una comunicación explícita entre objetos, con nombre y asíncrona) o una llamada (la invocación síncrona de una operación con un mecanismo para el control, que retorna posteriormente al remitente)[16].

La creación de un nuevo objeto se modela como un evento causado por el objeto creador y recibido por la propia clase. El evento de creación está disponible para la nueva instancia como evento actual en la transición desde el estado inicial del objeto.

El orden de los mensajes se puede mostrar en dos tipos de diagramas: un diagrama de secuencia (que se centra en las secuencias de tiempo de los mensajes) y un diagrama de colaboración (que se centra en las relaciones entre los objetos que intercambian los mensajes).

IV.2.1. Modelado con Diagrama de Secuencia

IV.2.1.1 Descripción

El diagrama de secuencia es más adecuado para observar la perspectiva cronológica de las interacciones entre objetos, muestra la secuencia explícita de mensajes y son mejores para especificaciones de tiempo real y para escenarios complejos. Representa una interacción, un conjunto de comunicaciones entre objetos organizados visualmente por orden temporal. A diferencia de los diagramas de colaboración, los diagramas de secuencia incluyen secuencias temporales pero no incluyen las relaciones entre objetos. Pueden existir de forma de descriptor (describiendo todos los posibles escenarios) y en forma de instancia (describiendo un escenario real). El diagrama de colaboración puede obtenerse automáticamente a partir del correspondiente diagrama de secuencia (o viceversa).

Dentro del conjunto de mensajes representados en una secuencia temporal, cada rol en la secuencia se muestra como una línea de vida, es decir, una línea vertical que representa el rol durante cierto plazo de tiempo, con la interacción completa. Los mensajes se muestran como flechas entre líneas de vida. Un diagrama de secuencia puede mostrar un escenario, es decir, una historia individual de transacción. Un uso de un diagrama de secuencia es mostrar la secuencia del comportamiento de un caso de uso, es por eso que se utilizan frecuentemente para validar cada caso de uso. Observando qué mensajes se envían a los objetos, componentes o casos de uso y analizando cuanto tiempo consume el método invocado, los diagramas de secuencia nos ayudan a comprender los cuellos de botella potenciales, para así poder eliminarlos.

Típicamente se examina la descripción de un caso de uso para determinar qué objetos son necesarios para la implementación del escenario. Si se tiene modelada la descripción de cada caso de uso como una secuencia de varios pasos, entonces se pueden seguir sobre ellos para descubrir qué objetos son necesarios para que se puedan seguir los pasos.

[16] El lenguaje unificado de modelado. Manual de referencia. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 2000.

Los conceptos más importantes relacionados con los diagramas de secuencia son:

- Línea de vida de un objeto (*lifeline*): Representa la vida del objeto durante la interacción. En un diagrama de secuencia un objeto se representa como una línea vertical punteada con un rectángulo de encabezado y con rectángulos a través de la línea principal que denotan la ejecución de métodos (activación). El rectángulo de encabezado contiene el nombre del objeto y el de su clase, en un formato *nombreObjeto*, como se muestra en la figura IV.7.



Figura IV.7. Representación de la línea de vida de un objeto

- Activación: Es la ejecución de un procedimiento, muestra el período de tiempo en el cual el objeto se encuentra desarrollando alguna operación, ya sea por sí mismo o delegándola en alguno de sus atributos. Se denota como un rectángulo delgado sobre la línea de vida del objeto, como se muestra en la figura IV.8.

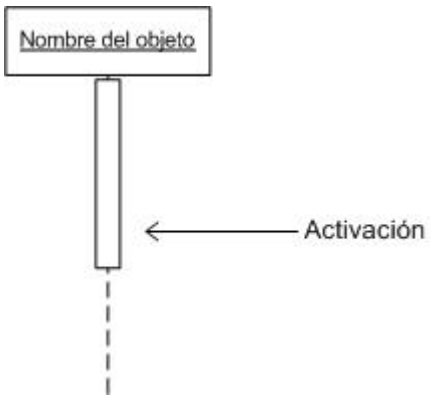


Figura IV.8. Representación de activación en un diagrama de secuencia

- 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, como se muestra en la figura, IV.9. Si un mensaje requiere un cierto tiempo para llegar a su destino, entonces la flecha del mensaje se dibuja diagonalmente hacia abajo.



Figura IV.9. Representación de mensaje

- Destrucción de un objeto: Se representa con una X grande en el punto en que deja de existir el objeto o en el punto en que el objeto se destruye a sí mismo.

- Llamada a un procedimiento u otro flujo de control anidado: La secuencia anidada completa debe finalizar antes de reanudar a la secuencia de nivel externo. Se puede emplear en llamadas normales a procedimiento. También se puede usar con objetos activos concurrentemente cuando uno de ellos envía una señal y espera a que finalice una secuencia de comportamiento anidada. Se representa con una flecha cuya punta está rellenada como se muestra en la figura IV.10.

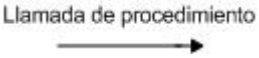


Figura IV.10. Representación de una llamada a un procedimiento u otro flujo de control anidado (call)

- Métodos recursivos: Si el objeto se llama a sí mismo, entonces se representa con una línea que regresa a la activación principal y con líneas de llamada de mensajes, que indican la entrada y salida de la recursión, como se muestra en la figura IV.11.



Figura IV.11 Representación de método recursivo (call)

- Retorno de una llamada a procedimiento: Se representa con una flecha discontinua con punta sin relleno como se muestra en la figura IV.12. La flecha de retorno puede suprimirse, por cuanto queda implícita al final de la activación.

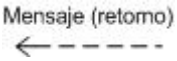


Figura IV.12. Representación de un mensaje de retorno

En la figura IV.13 se muestra una representación de un diagrama de secuencia, en el cual se pueden observar los elementos relacionados con este.

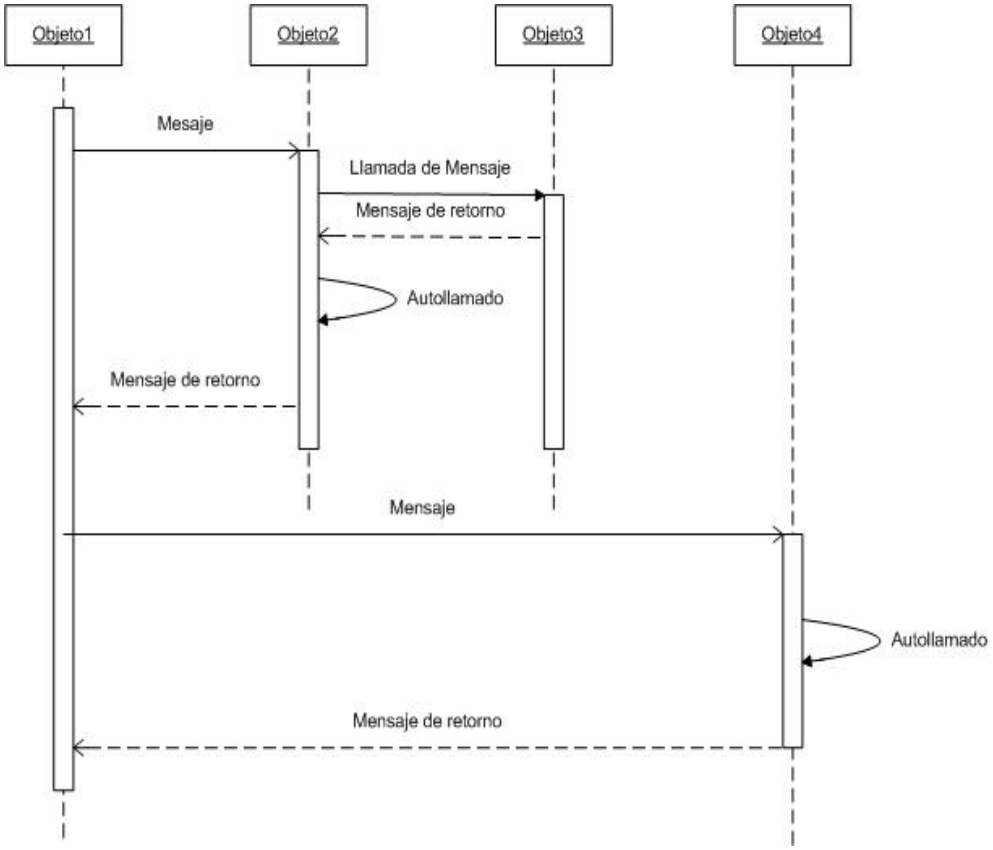


Figura IV.13. Representación de un Diagrama de Secuencia

Un diagrama de secuencia también se puede mostrar en forma de descriptor, en el cual los constituyentes son roles en lugar de objetos. Este diagrama muestra en el caso general, no una sola ejecución del mismo.

IV.3. Modelado con Diagrama de Clases

IV.3.1. Descripción de clase

El diagrama de clases es el diagrama principal para el análisis y diseño. Un diagrama de clases sirve para visualizar las relaciones entre las clases que involucran el sistema, las cuales pueden ser asociativas, de herencia, de uso y de contenimiento.

Un diagrama de clases esta compuesto por los siguientes elementos:

- Clase: atributos, métodos y visibilidad.
- Relaciones: Herencia, Composición, Agregación, Asociación y Uso.

Una clase es la unidad básica que encapsula toda la información de un objeto (un objeto es una instancia de una clase). A través de ella podemos modelar el entorno en estudio.

Las clases son una especificación de propiedades y funcionalidades de elementos que van a participar en nuestros procesos, puede representar: personas o roles, eventos y catálogos. Definen el negocio, representan conceptos que describen el entorno y pueden traslaparse.

La representación de la clase es un rectángulo con tres divisiones internas, los cuales son los elementos fundamentales del diagrama. Los tres compartimientos estándares alojan el nombre de la clase, sus atributos (o variables de instancia) que caracterizan a la clase, y sus métodos u operaciones, los cuales son la forma como interactúa el objeto con su entorno. Una representación gráfica de clase es la que se muestra en la figura IV.14.

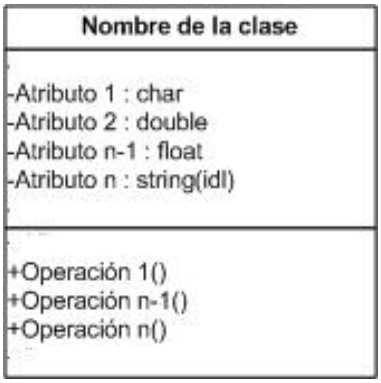


Figura IV.14. Representación de una clase

Es importante definir lo que es una subclase y una superclase debido a que estas se encuentran íntimamente ligadas al concepto de clase. Una subclase (clase derivada), es la descripción de una clase basada en la estructura de otra clase, la cual se denomina superclase (clase base). Una subclase se puede mirar como una extensión de la superclase, donde se pueden tomar las características de la superclase, y adicionarle nuevas características. Este concepto suele confundirse con el concepto de herencia, pero no siempre puede decirse que una subclase es una herencia.

Las clases se relacionan entre sí a través de las relaciones que son las líneas rectas entre dos clases y constan de roles y *cardinalidad*. Los roles son las frases que se encuentran en la relación y sirven para definir la cardinalidad de la relación, siempre considerando la unidad en la clase origen.

Como se mencionó anteriormente, los atributos identifican las características propias de cada clase. Generalmente son de tipos simples, ya que los atributos de tipos compuestos se representan mediante asociaciones de composición con otras clases.

El conjunto de operaciones describen el comportamiento de los objetos de una clase. Existen operaciones que se implementan de manera distinta en cada

subclase. Estas operaciones reciben el nombre de polimórficas. Cuando existen versiones polimórficas de una operación, debe haber una especificación por cada versión.

Tanto los atributos como los métodos cuentan con una sintaxis. Para los atributos la sintaxis en UML es la siguiente[17]:

visibilidad nombre : tipo = valor_inicial {lista_de_posibles_valores}

y para los métodos, la sintaxis es la siguiente:

visibilidad nombre (lista_parámetros) : tipo_de_retorno
{lista_de_posibles_valores}

donde la lista de parámetros consta de la siguiente sintaxis:

nombre : tipo = valor_default

Los parámetros van separados por comas y la visibilidad tanto para los atributos como para los métodos, puede ser público o privado debido a que en una clase no deben ser manipulables directamente por el resto de los objetos. Por esta razón se crearon niveles de visibilidad para los elementos como se muestra a continuación.

Para los atributos o características, así como para los métodos u operaciones de una clase son:

- public (+): Indica que el atributo o método será visible tanto dentro como fuera de la clase, es decir, es accesible desde todos lados.
- private (-): Indica que el atributo o método sólo será accesible desde dentro de la clase (sólo sus métodos lo pueden accesar).
- protected (#): Indica que el atributo o método no será accesible desde fuera de la clase, pero si podrá ser accesado por métodos de la clase además de las subclases que se deriven (herencia).

Los enlaces entre objetos pueden representarse entre las respectivas clases y sus formas de relación son:

- Asociación y agregación (vista como un caso particular de asociación)
- Generalización/Especialización.

Las relaciones de agregación y generalización forman jerarquías de clases. Este tipo de relaciones así como su representación están definidas en el Capítulo III.

La noción de clase está próxima a la de conjunto. Dada una clase, podemos ver el conjunto relativo a las instancias que posee o bien relativo a las propiedades de la clase. La generalización expresa relación de inclusión entre conjuntos.

El modelo de casos de uso aporta información para establecer las clases, objetos, atributos y operaciones. Para la realización del diagrama de clases también se toman como base los diagramas de secuencia y de colaboración por lo que se manejarán los objetos que ahí se consideraron pero ahora a nivel de clases.

[17] http://www.dcc.uchile.cl/~psalinas/uml/modelo.html

Además, se pueden agregar nuevas clases que no se habían considerado. Para poder definir las clases, UML sugiere seis características selectivas que debe utilizar el analista para considerar una clase candidato en el modelo de análisis:

1. Información retenida. La clase será útil durante el análisis sólo si la información sobre el mismo ha de ser almacenada, transformada, analizada o manejada en algún otro modo. La información puede referirse a conceptos que deberán estar siempre registrados en el sistema, eventos o transacciones que ocurren en un momento específico.

2. Sistema externo. Si se tiene un sistema externo a este sistema, entonces es de interés en la etapa de modelado. Los sistemas externos deberán ser vistos como clases que el sistema contendrá o con los cuales interactuará.

3. Patrones, librerías de clases o componentes. Si se tienen patrones, librerías de clases o componentes, generalmente éstos son clases candidatos.

4. Dispositivos que el sistema maneja. Dispositivos técnicos que maneja el sistema se convertirán en clases que manejarán esos dispositivos.

5. Partes organizacionales. Especialmente en modelos de negocio, todas las partes que representan a la organización, serán clases candidatos.

6. Roles de actores. Los roles de actores serán vistos como clases, por ejemplo, usuario, operador del sistema, administrador, cliente, etc.

IV.4. Modelado con Diagrama de Paquetes

IV.4.1. Descripción de paquete

Cualquier sistema grande se debe dividir en unidades más pequeñas, de modo que las personas puedan trabajar con una cantidad de información limitada, a la vez y de modo que los equipos de trabajo no interfieran con el trabajo de los otros.

Un paquete es una parte de un modelo. Cada parte del modelo debe pertenecer a un paquete. Pero para ser funcional, la asignación debe seguir un cierto principio racional, tal como funcionalidad común, implementación relacionada y punto de vista común. UML no impone una regla para componer los paquetes.

Se dibujan como rectángulos con pestañas (similar al icono "carpeta"), como se muestra en la figura IV.15, las dependencias se muestran como flechas con líneas discontinuas.

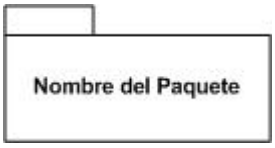


Figura IV.15. Representación de un paquete

Los paquetes ofrecen un mecanismo general para la organización de los modelos/subsistemas agrupando elementos de modelado. Cada paquete corresponde a un submodelo (subsistema) del modelo (sistema). Los paquetes son unidades de organización jerárquica de uso general de los modelos de UML. Pueden ser utilizados para el almacenamiento, el control de acceso, la administración de la configuración y la construcción de bibliotecas que contengan fragmentos reutilizables del modelo[18].

Un subsistema es un paquete que tiene piezas separadas de especificación y de realización. Representa una unidad coherente del modelo, con interfaces limpias al resto del sistema. Representa generalmente la partición del sistema. Los modelos y los subsistemas se dibujan como paquetes con las palabras clave de estereotipo, como se muestra en la figura IV.16.

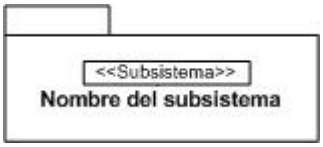


Figura IV.16. Representación de un paquete

Un paquete puede contener otros paquetes, sin límite de anidamiento pero cada elemento pertenece a (*está definido en*) sólo un paquete.

Los paquetes contienen elementos del modelo al más alto nivel, tales como clases y sus relaciones, máquinas de estado, diagramas de casos de uso, interacciones y colaboraciones; atributos, operaciones, estados, líneas de vida y mensajes están contenidos en otros elementos y no aparecen como contenido directo de los paquetes.

Las dependencias que se presentan entre elementos individuales, pero en un sistema de cualquier tamaño, deben ser vistas en un nivel más alto. Las dependencias entre paquetes resumen dependencias entre los elementos internos a ellos, es decir, las dependencias del paquete son derivables a partir de las dependencias entre los elementos individuales.

La presencia de una dependencia entre paquetes implica que existe en un enfoque ascendente (una declaración de existencia), o que se permite que exista más adelante en un enfoque descendente (una restricción que limita cualquier otra relación), por lo menos un elemento de relación con el tipo de dependencia indicado entre elementos individuales dentro de los paquetes correspondientes.

Las dependencias múltiples del mismo tipo entre elementos individuales se agregan a una sola dependencia entre los paquetes que contienen los elementos. Si las dependencias entre elementos contienen estereotipos, éste puede ser omitido en la dependencia del paquete, para dar una sola dependencia de alto nivel.

[18] UML Gota a gota. M. Fowler, K. Scott. Ed. Prentice Hall. 1999.

Una clase de un paquete puede aparecer en otro paquete por la importación a través de una relación de dependencia entre paquetes. Todas las clases no son necesariamente visibles desde el exterior del paquete, es decir, un paquete encapsula a la vez que agrupa.

En la figura IV.17 se muestra un diagrama de paquetes en el que se puede observar la representación del sistema y subsistemas así como sus dependencias.

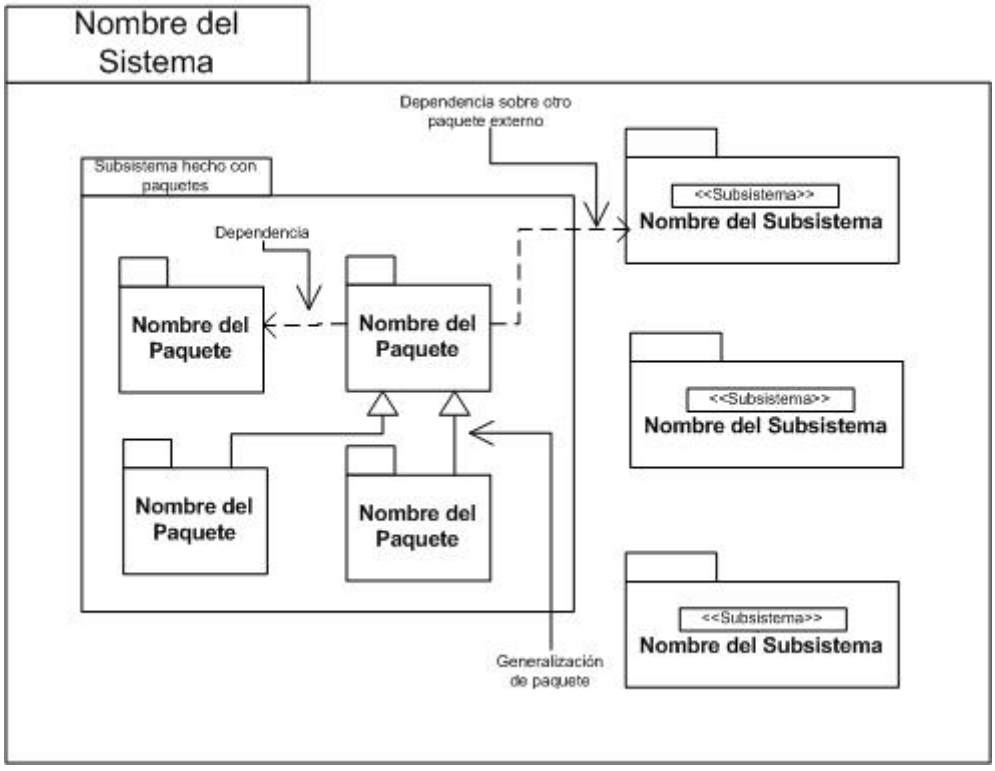


Figura IV.17. Representación de un diagrama de paquetes

En general, un paquete no puede tener acceso al contenido de otro paquete. Los paquetes son opacos, a menos que sean abiertos por una dependencia de acceso o de importación. La dependencia de acceso indica que el contenido del paquete del proveedor puede aparecer en referencias efectuadas por los elementos del paquete cliente. Generalmente, un paquete puede ver solamente los elementos de otros paquetes que tienen visibilidad pública. Los elementos con visibilidad protegida pueden ser vistos únicamente por los paquetes que son descendientes del paquete contenedor de dichos elementos. Los elementos con visibilidad privada sólo son vistos por su paquete contenedor y anidados. El permiso de acceso y visibilidad son necesarios para hacer referencia a un elemento.

DEFINICIÓN DE LA ARQUITECTURA

V. DEFINICIÓN DE LA ARQUITECTURA

Gran parte del modelo de un sistema se piensa para mostrar los aspectos lógicos y diseño del sistema, independiente de su empaquetado final en el cual se desarrolló. Sin embargo, los aspectos de implementación son importantes para los propósitos de reutilización y del rendimiento. UML incluye dos vistas para representar unidades de implementación: la vista de implementación y la vista de despliegue.

La vista de implementación muestra el empaquetado físico de las partes reutilizables del sistema en unidades sustituibles, llamadas componentes. Este tipo de vista muestra la implementación de los elementos del diseño (tales como clases) mediante componentes, así como sus interfaces y dependencias entre estos. Los componentes son las piezas reutilizables de alto nivel a partir de las cuales se pueden construir los sistemas.

La vista de despliegue muestra la disposición física de los recursos de ejecución computacional, tales como computadoras y sus interconexiones, los cuales se llaman nodos. Durante la ejecución, los nodos pueden contener componentes y objetos. La asignación de componentes y de objetos a los nodos puede ser estática, o pueden migrar entre nodos.

La arquitectura es el conjunto de decisiones significativas sobre la organización de un sistema de software. Incluye la selección de elementos estructurales y las interfaces mediante las que se conectan, la organización a gran escala de los elementos, los mecanismos importantes de que se dispone en el sistema y el estilo arquitectónico que guía su organización.

Las decisiones arquitectónicas sobre la descomposición de un sistema en partes se pueden capturar utilizando modelos, subsistemas, paquetes, y componentes. Las dependencias entre estos elementos son indicadores clave de la flexibilidad de la arquitectura y de la dificultad de modificar el sistema en el futuro.

V.1. Descripción de los componentes

Un componente es una unidad física de implementación con interfaces bien definidas pensada para ser utilizada como parte reemplazable de un sistema. Cada uno de ellos incorpora la implementación de ciertas clases del diseño del sistema. Los componentes bien diseñados no dependen directamente de otros sino de las interfaces que ofrecen estos[19].

Algunos componentes tienen identidad y pueden poseer entidades físicas, que incluyen objetos en tiempo de ejecución, documentos, bases de datos, etc. Los componentes existentes en el dominio de la implementación son unidades físicas en las computadoras que se pueden conectar con otros, sustituir, trasladar, archivar, etc.

[19] El lenguaje unificado de modelado. Manual de referencia. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 2000.

Un componente se representa mediante un rectángulo, con dos rectángulos pequeños a un lado como se muestra en la figura V.1. Lo cual representa un empaquetamiento físico.



Figura V.1. Representación gráfica de un componente

Los componentes tienen dos características: Empaquetan el código que implementa la funcionalidad de un sistema y algunas de sus propias instancias de objetos que constituyen el estado del sistema. Los últimos llamados componentes de la identidad, ya que sus instancias poseen identidad y estado.

Un componente de código fuente es un paquete para el código fuente de las clases de implementación. Algunos lenguajes de programación distinguen archivos de declaración de los archivos de método, pero todos son componentes. Un componente de código binario es un paquete para el código compilado. Una biblioteca del código binario es un componente.

Cada tipo de componente contiene el código para las clases de implementación que realizan algunas clases e interfaces lógicas. La relación de realización asocia un componente con las clases y las interfaces lógicas que comprenden sus clases de implementación. Las interfaces de un componente describen la funcionalidad que aporta. Cada operación de la interfaz debe hacer referencia eventualmente a un elemento de la implementación disponible en el componente.

La estructura estática ejecutable de una implementación de un sistema, se puede representar como un conjunto interconectado de componentes. Las dependencias entre estos significan que los elementos de la implementación en un componente requieren los servicios de los elementos de implementación en otros. Tal uso requiere que dichos elementos sean de visibilidad pública.

Las dependencias de un componente con otros o con elementos del modelo se representan usando líneas discontinuas con la punta de flecha hacia los elementos del proveedor, como se muestra en la figura V.2.

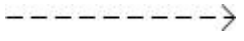


Figura V.2. Representación de dependencia de un componente

Un componente ofrece un conjunto de elementos de implementación, esto significa que el componente proporciona el código para los elementos. Puede tener operaciones e interfaces. Un componente de identidad es un contenedor físico para

las entidades físicas como bases de datos. Para proporcionar manejadores para sus elementos contenidos, puede tener atributos y asociaciones salientes, que deben ser realizadas por sus elementos de implementación.

Sí un componente es la realización de una interfaz, se representa con un círculo unido al símbolo del componente por un segmento de línea, como se muestra en la figura V.3.

Figura V.3. Representación de una interfaz

Una interfaz es una lista de las operaciones que una pieza de software o de hardware ofrece y puede realizar. El uso de las interfaces permite evitar las dependencias directas entre componentes, facilitando una sustitución más fácil de nuevos componentes. Las interfaces son los puntos visibles de entrada o los servicios que un componente está ofreciendo y dejando disponibles a otros componentes de software y clases.

El diagrama de componentes muestra la red de dependencias entre estos y pueden aparecer de dos formas. Pueden mostrar un conjunto de componentes disponibles (una biblioteca de componentes) con sus dependencias; éste es el material a partir del cual se puede ensamblar un sistema o puede también mostrar un sistema configurado, con la selección de componentes usados para construirlo.

Los diagramas de componentes se utilizan para modelar la vista estática de un sistema. Lo que los distingue de otros tipos de diagramas es su contenido. Normalmente contienen componentes, interfaces y relaciones entre ellos. En el se sitúan librerías, tablas, archivos, ejecutables y documentos que formen parte del sistema. Uno de los usos principales es que puede servir para ver que componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema. Y como todos los diagramas, también puede contener paquetes utilizados para agrupar elementos del modelo[20].

Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre los componentes de *software*, sean ya componentes de código fuente, binarios o ejecutables. Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del *software*, la reutilización, sus interfaces, interrelaciones y las restricciones impuestas por los lenguajes de programación, así como las herramientas utilizadas en el desarrollo. Los elementos de modelado dentro de un diagrama de componentes serán componentes y paquetes.

Cada componente en el diagrama debe ser documentado con un diagrama de componentes más detallado, un diagrama de clases o un diagrama de casos de uso.

Un diagrama de componentes se representa como un esquema de componentes *software* unidos por medio de relaciones de dependencia (generalmente de compilación). Puede mostrar también que un componente *software* contiene una interfaz, es decir, la soporta. En la figura V.4 se muestra la representación de un diagrama de componentes.

[20] El lenguaje unificado de modelado. Manual de referencia. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 2000.

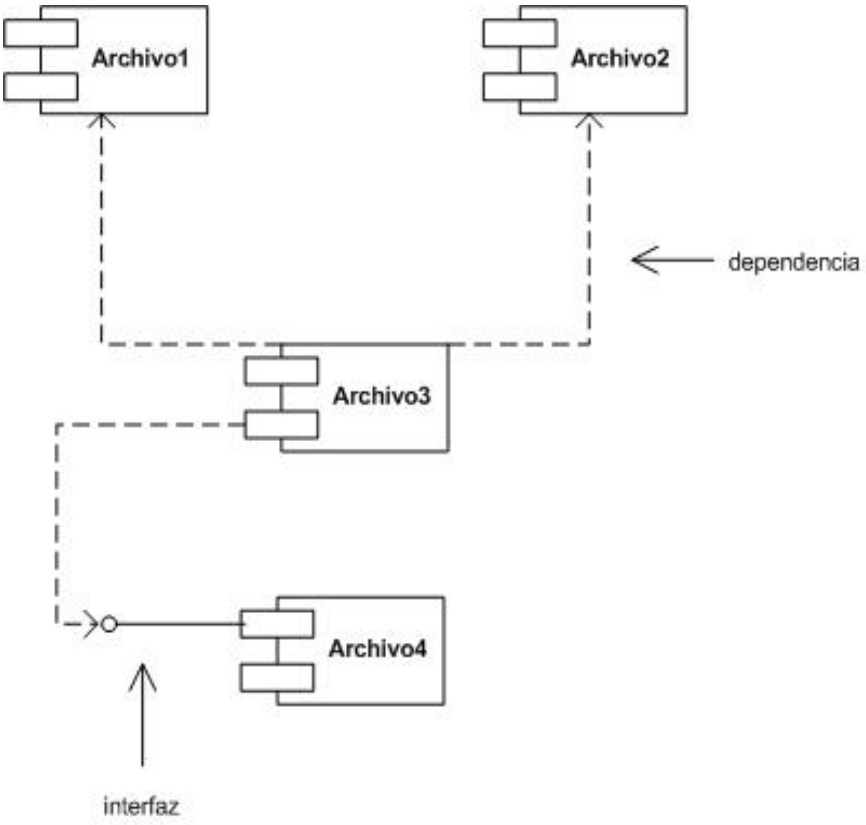


Figura V.4. Representación gráfica de un diagrama de componentes

No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema.

Normalmente los diagramas de componentes se utilizan para modelar código fuente, versiones ejecutables, bases de datos físicas, entre otros[21]:

- Código fuente: En el modelado de código fuentes se suelen utilizar para representar las dependencias entre los archivos de código fuente, o para modelar las diferentes versiones de estos archivos. Para ello se deben identificar el conjunto de archivos de código fuente de interés y estereotiparlos como archivos file, agruparlos en paquetes, utilizar valores etiquetados para la información de versiones y modelar las dependencias de compilación.

- Código ejecutable: Se utiliza para modelar la distribución de una nueva versión a los usuarios. Para tal propósito se identifican el conjunto de componentes ejecutables que intervienen, se utilizan estereotipos para los diferentes tipos de componentes (ejecutables, bibliotecas, tablas, archivos, documentos, etc.), se consideran las relaciones entre dichos componentes

[21] El lenguaje unificado de modelado. J. Rumbaugh, I. Jacobson, G. Booch. Ed. Pearson, 1999.

que la mayoría de las veces incluirán interfaces que son exportadas por ciertos componentes e importadas por otros.

- Bases de datos física: UML permite el modelado de bases de datos físicas, así como de los esquemas lógicos de estas.

Así si tenemos en cuenta las dependencias asociadas al proceso de compilación un componente podría ser:

- Código fuente que depende de otros componentes que deben estar disponibles durante la compilación del componente. Se utiliza para documentar las dependencias de los diferentes archivos de código fuente. Un ejecutable o librería es una combinación de estos archivos y al mostrar la dependencia entre ellos obtenemos una visión de las partes necesarias para la creación del ejecutable o librería. El tener documentadas las relaciones se pueden realizar cambios en el código de un archivo teniendo en cuenta donde se utiliza y que otros archivos pueden verse afectados por su modificación.

- Código objeto binario, como por ejemplo una librería, que puede depender de algún código objeto con el que se liga.

- Código ejecutable, puede depender de otros programas ejecutables con los que interacciona en tiempo de ejecución. En la figura V.5 se ilustra un diagrama de componentes en el que se muestra como un ejecutable utiliza dos librerías, estas disponen de su interfaz con el que se tiene el acceso a sus servicios. Se puede ver que estas librerías son componentes que pueden ser reutilizados en otras partes del sistema.

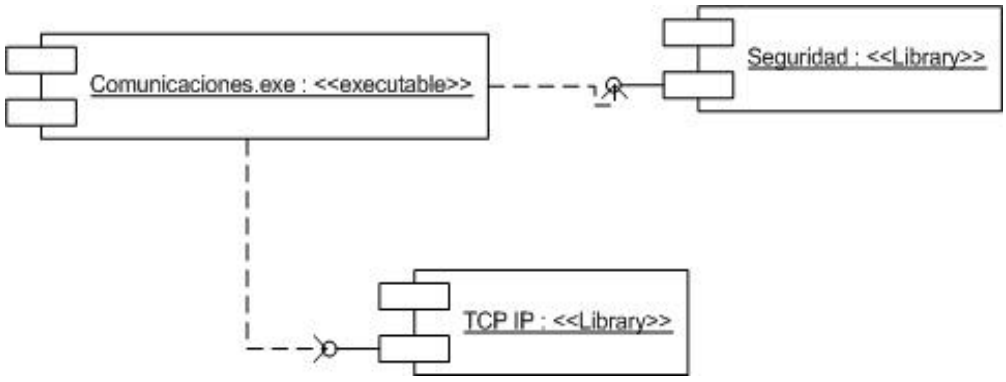


Figura V.5. Representación del uso de interfaces

Se pueden utilizar estereotipos como «link» o «compile» para distinguir la distinta naturaleza de las dependencias. Igualmente se pueden definir estereotipos para las distintas clases de componentes. UML proporciona algunos estereotipos predefinidos como:

- «file»
- «library»
- «executable»
- «table»

V.2. Modelado con Diagramas de Despliegue

Un nodo es un objeto físico de ejecución que representa un recurso computacional, que generalmente tiene por lo menos memoria y a menudo también capacidad de proceso. Los nodos pueden tener estereotipos para distinguir diferentes tipos de recursos, tales como CPU, dispositivos y memorias. Los nodos pueden contener objetos, instancias e instancias del componente[22].

Un nodo se representa mediante un cubo con su respectivo nombre, y opcionalmente su clasificación. La figura V.6 muestra la representación de un nodo.



Figura V.6. Representación de un nodo

En ésta etapa de desarrollo se describe la configuración del sistema para su ejecución en un ambiente del mundo real. Para el despliegue se deben tomar decisiones sobre los parámetros de la configuración, funcionamiento, asignación de recursos, distribución y concurrencia. Los resultados de esta fase se capturan en archivos de configuración y también en diagramas de despliegue.

Un diagrama de despliegue muestra la distribución física del sistema en un ambiente de producción (o de prueba). Muestra dónde se ubican los componentes, en qué servidores, máquinas o hardware. Puede representar los enlaces de redes, el ancho de banda de la LAN, etc. En la figura V.7 se muestra la representación de un diagrama de despliegue en el cual se pueden ver los nodos, sus relaciones y donde cada nodo está a su vez formado por componentes.

[22] UML Gota a gota. M. Fowler, K. Scott. Ed. Prentice Hall. 1999.

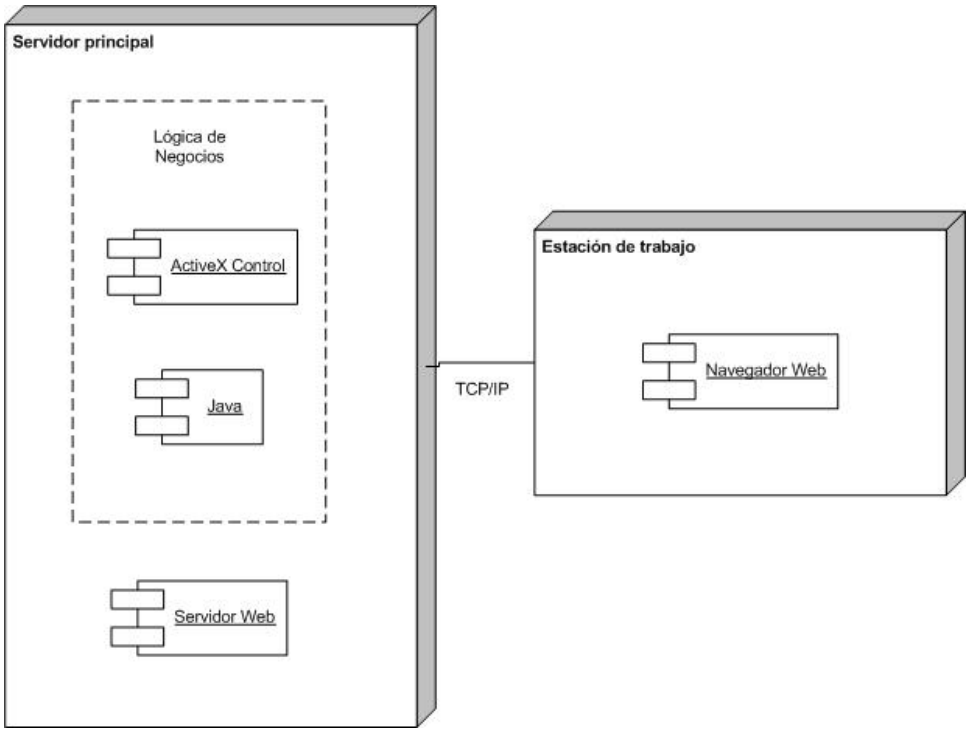


Figura V.7. Representación de un diagrama de despliegue

Los componentes pueden tener requisitos adjuntos para indicar sus obligaciones contractuales; es decir, qué servicios se proveen en el modelo. Los requisitos ayudan a documentar el comportamiento funcional de los elementos de software.

También pueden tener restricciones asignadas que indican el entorno en el que operan. Las pre-condiciones especifican lo que debe ser verdadero antes de que un componente pueda realizar alguna función; las post-condiciones indican lo que debe ser verdadero después de que un componente haya realizado algún trabajo y los invariantes especifican lo que debe permanecer verdadero durante la vida del componente.

Los escenarios son descripciones textuales y procedimentales de las acciones de un objeto a lo largo del tiempo y describen la forma en la que un componente trabaja. Se pueden crear múltiples escenarios para describir tanto el camino básico (una ejecución perfecta), como las excepciones, errores y otras condiciones.

La traza es una dependencia que indica una conexión entre dos elementos que representan el mismo concepto en dos niveles diferentes de significado. No representa una semántica dentro de un modelo. Más bien, representa conexiones entre elementos de semántica diferente, de elementos pertenecientes a diferentes modelos situados en distintos planos de significado.

MODELADO GENERAL DE PROCESOS DE VENTAS PARA EMPRESAS

VI. MODELADO GENERAL DE PROCESOS DE VENTAS PARA EMPRESAS

VI.1. Descripción del modelo

La empresa base para el desarrollo del modelo es una empresa fundada en 1997, inició su trayectoria especializada en sistemas UNIX, abasteciendo al mercado mexicano un portafolio de productos y servicios alrededor de esta plataforma. Con la evolución de las tecnologías han incorporado a su portafolio de negocios una serie de fabricantes que cuentan con productos innovadores que solucionan problemáticas de las empresas actuales.

Su estrategia empresarial es ofrecer soluciones así como una plataforma para negocios a sus clientes. Es por ello que cuentan con un área de servicios donde se cubren las necesidades más importantes, como son: soporte técnico, capacitación y servicios profesionales.

A medida que la empresa crece, los procesos administrativos se han vuelto complejos y difíciles de controlar sin una herramienta informática.

Este es el caso del área de ventas, la cual ha tenido un crecimiento en el número de productos que actualmente maneja la empresa y por lo tanto, ha aumentado el número de vendedores, los cuales no tienen ningún patrón a seguir para el proceso de ventas.

La razón principal por las que la empresa esta considerando una nueva implementación de un sistema CRM (Customer Relationship Management), con el que pueda mejorar su proceso de ventas, es debido a que no cuentan con una herramienta con la que la pueda administrar su información, ya que sus procesos no han sido establecidos. Esto implica que no tiene patrones de referencia para medir el trabajo de sus empleados, por lo tanto no saben si el trabajo que están realizando es el correcto o si este puede ser mejorado para lograr un mejor desempeño de la empresa.

Cabe mencionar que actualmente la empresa cuenta con un CRM, al cual llamaré CRM-A, que no fue hecho a la medida, por lo que no es funcional, y el uso de esa herramienta es mucho más complicada que usar los métodos con los que contaban anteriormente para manejar su información. El método anterior al CRM-A actual para administrar la información de ventas, constaba de mantener su información en una hoja de cálculo en la que cada vendedor llevaba el control de su información como mejor la entendía y en orden de importancia de acuerdo al punto de vista de cada vendedor, aunque este método era muy sencillo, el buscar información era mucho más ágil puesto que el vendedor tenía ubicada la información a encontrar mucho más fácil y rápido que con el uso del CRM-A.

El CRM-A está diseñado para mostrar reportes de ventas tales como: número de ventas del vendedor, que producto es el más vendido, que ingeniero da mayor número de soportes, cual es el cliente que compra más, promedio de ventas mensuales, venta total mensual, etc. Desafortunadamente estos reportes del CRM-A no son del todo confiables, puesto que como no fue hecho a la medida, hay mucha información que no puede introducirse a la base de datos del CRM-A, lo que ocasiona que la información no este completa y por ende los reportes tampoco lo están.

A grandes rasgos la empresa está dividida en seis áreas laborales, las cuales son las siguientes:

- Ventas: Área que se encarga de vender el software, hardware, soporte técnico, consultoría y capacitación. Para cada producto existe un vendedor dedicado, quien es el encargado de llevar el control de clientes, y darle seguimiento a los proyectos de pre-venta, venta y post-venta. Los vendedores son los encargados de realizar cotizaciones, darle seguimiento a los proyectos, realizar órdenes de facturación y órdenes de envío para el envío de los productos. Estas órdenes deben ser entregadas al área administrativa para que proceda el siguiente proceso realizado por esa área.

- Soporte técnico: Área que se encarga de dar el soporte técnico para los productos que se venden en la empresa, además de dar servicio de consultoría y capacitación. Está íntimamente ligado con el departamento de ventas puesto que muchas veces los productos se venden con servicios de instalación y puesta a punto o pólizas de soporte. Estas áreas deben estar coordinadas para llevar el control de la disponibilidad de los Ingenieros de soporte, ya que la consultoría, soporte técnico y capacitación, es un servicio que generalmente se vende independiente de los productos, por lo que pueden ser los mismos clientes de productos o únicamente clientes de esta área.

- Administración y contabilidad: Área encargada de compras, cuentas por pagar, cuantas por cobrar, facturación, envío de productos, inventario, nómina y contabilidad. Por tal motivo está muy ligado con el área de ventas para dar el seguimiento de cobranza de los productos vendidos y facturación. Así como del envío correcto y a tiempo de los productos, debido a que sin el pago de los productos, éstos no pueden enviarse a su destino. Esta área también maneja el inventario de productos en almacén, por lo que cuando alguno de ellos no se tiene o está por terminarse, el vendedor tiene que contactar a la persona encargada de almacén e inventario para la solicitud de los mismos. En caso de que el cliente necesite algún tipo de crédito, esta área es la que se encarga de solicitar los documentos necesarios y del análisis para la aceptación del crédito.

- Sistemas: Área encargada de los sistemas informáticos y telecomunicaciones internos. Entre sus actividades se encuentra el dar mantenimiento preventivo y correctivo de PC's, administración de servidores en producción como lo es el de correo electrónico y página Web, así como la verificación de que éstos siempre estén trabajando correctamente. Se encargan del control y administración de software y hardware dentro de la empresa. Esta área está ligada con la de ventas, puesto que la mayoría de las cotizaciones son enviadas por correo electrónico y para el seguimiento de estas es muy utilizado también. Por otra parte el área de soporte técnico tiene pólizas de soporte técnico vía e-mail, por lo que es necesario que este funcione correctamente y siempre esté en producción. Además de que el servidor que aloja la página Web debe de estar trabajando correctamente porque muchos de los clientes llegan por la información que obtienen de la página.

- Marketing: Área encargada de promover los productos que se comercializan en la empresa. Esta promoción se hace a través de eventos que se organizan cada cierto tiempo generalmente para lanzar nuevas versiones de producto o para dar a conocer los productos. A parte de ser un área muy ligada con la de ventas debido a que por medio de la promoción los productos se dan a conocer y las ventas aumentan, es muy importante

porque en todos los eventos que organiza se realizan cuestionarios de evaluación, los cuales son una retroalimentación para los vendedores, ya que contienen opinión sobre los productos y sobre la atención al cliente.

- Recursos humanos: Área encargada de la selección, evaluación y contratación de empleados, elaboración de contratos, generación de ofertas de empleo. Es el área que lleva la administración de la información de los empleados como es su historial familiar, académico y laboral.

La forma en que se comercializan los productos o servicios es a través del contacto directo con el cliente o a través de distribuidores. Los clientes pueden ser clientes que ya se tienen registrados en la cartera o nuevos clientes. Los distribuidores pueden ser especiales, certificados o registrados, este nombramiento depende de la relación de la empresa con el distribuidor y por ende, de los compromisos y el descuento que se otorga a cada uno de ellos.

Generalmente la venta a clientes que ya se tienen registrados, es porque éstos contactan a la empresa o al vendedor vía telefónica o e-mail, ya que necesitan un producto más de los que ya cuenta o alguna actualización del producto con el que está trabajando o simplemente para saber si cuentan con una solución que pueda ayudarle en su trabajo. De este tipo de cliente se tiene al menos la información necesaria para contactarlo, debido a que ya hubo alguna relación anterior con ellos ya sea de alguna cotización o venta, es por ello que se cuenta al menos con la información necesaria como es: nombre de la empresa, nombre del contacto, teléfono y e-mail, de algunos otros clientes se cuenta con información más completa como su dirección y algunos de los productos que ya ha comprado. A partir de la solicitud de información sobre el producto, el vendedor hace la cotización y empieza a darle seguimiento a la venta.

La obtención de nuevos clientes es generalmente a través de llamadas telefónicas realizadas por ellos, porque necesitan o tienen interés en algún producto de los que comercializa la empresa.

La página Web juega un punto muy importante puesto que la mayoría de los nuevos clientes contactan por la información que obtienen de la página. Generalmente este tipo de venta se realiza cuando el cliente contacta al vendedor vía telefónica o e-mail, porque ya tiene conocimiento del producto que necesita ó para preguntar si existe alguna solución que pueda ayudarle. A partir de ese momento se empieza a darle seguimiento a la venta por lo que es importante capturar los datos del cliente.

La venta a través de distribuidores es muy similar a la venta con usuario final. La diferencia reside en el tipo de distribuidor, que como ya se mencionó hay tres tipos de distribuidores y de ello dependerá el descuento que se otorgue a cada unos de ellos. El distribuidor especial, es aquel que distribuye el producto de manera constante, busca nuevos clientes y proyectos, cuenta con certificaciones del producto, se actualiza constantemente y la relación de trabajo es muy estrecha con la empresa, por ello el descuento que se les otorga es mayor que a los otros dos tipos de distribuidores. Los distribuidores certificados son aquellos que ya han tomado cursos de certificación con la empresa, sin embargo su trabajo de venta no es tan constante por lo que el descuento es un poco menor. El distribuidor registrado es aquel que no es constante y solamente vende el producto cuando algún cliente se lo solicita, por lo tanto la relación y el compromiso que se tienen con la empresa es casi nulo, por lo que el descuento disminuye. Generalmente se cuenta con la información necesaria de estos tipos de distribuidores, ya que al menos deben ser registrados en un documento y por lo tanto se tiene información como: nombre de la empresa, contactos, teléfono, e-mail, RFC, domicilio.

VI.2. Aplicación del modelo

Después de platicar con los vendedores y con las personas relacionadas al proceso de ventas, a grandes rasgos se puede decir que actualmente la empresa cuenta con los siguientes módulos relacionados con su proceso de ventas, cabe mencionar que estas actividades son realizadas pero no existe ningún documento que indique como deben de realizarse, por lo que las actividades son realizadas de acuerdo al criterio de la persona que lo está llevando a cabo, lo que no garantiza que el proceso esté bien hecho y sea efectivo, ni tampoco que todas las personas lo lleven a cabo. De igual manera, cada vez que una persona de ventas es contratada, no hay un proceso escrito para entrenarlo sobre el proceso de ventas, por lo que el entrenamiento es únicamente comentado, y con esto se corre el riesgo de que pasen desapercibidos muchos aspectos importantes para el proceso.

Como se menciona en el Capítulo III, para que pueda hacerse un monitoreo efectivo del proceso que se lleva a cabo es importante tener claras las tareas que se llevan a cabo, y en base a esto, asignar responsables para su correcto funcionamiento y para asegurar que ninguna tarea deje de realizarse por pequeña que esta parezca.

A continuación se muestra la tabla 1, en la que se muestran los actores, tareas y responsabilidades de acuerdo al análisis de la empresa.

Actor	**Tareas**	**Responsabilidades**
Director comercial	Establecer relaciones de negocio	Establecer relaciones comerciales con distribuidores
		Establecer relaciones comerciales con proveedores
	Administrar fuerzas de ventas	Seleccionar vendedores
		Planear capacitaciones internas de los productos
		Revisar evaluaciones de venta
	Analizar costo-beneficio	Revisar el pipeline
		Revisión el forecast
		Revisar el reporte de ventas
		Establecer cuotas
	Autorizar actividades	Autorizar descuentos en cotizaciones
		Autorizar promociones
		Autorizar pedidos
		Autorizar eventos
Vendedor de pre-venta	Mantener la base instalada de clientes	Obtener nuevos clientes
		Visitar a clientes
		Enviar información electrónica a clientes
		Enviar información impresa a clientes
	Organizar pruebas de concepto	Dar a conocer los requerimientos
		Dar a conocer la forma de operación del producto
	Analizar la pre-venta	Pasar la información al gerente de producto para la realización de la venta

		Llevar la información de pipeline
Gerente de producto	Administrar información clientes	Obtener información de clientes
		Corroborar la información que se tiene
		Mantener actualizada la base instalada de clientes
	Cotizar	Hacer cotizaciones
		Enviar cotizaciones a clientes
	Realizar actividades administrativas	Elaborar orden de facturación
		Solicitar obtener documentos de facturación al asistente
		Solicitar factura al encargado de facturación
		Elaborar orden de pedido interno
		Solicitar pedido al encargado de compras
		Llevar inventario de producto que maneja
	Elaborar plan de trabajo	Hacer listas de precios
		Hacer plan de trabajo
	Planear actividades	Establecer promociones
		Proponer promociones
		Proponer eventos
		Solicitar autorización de actividad
	Reportar de ventas	Elaborar reporte de ventas
		Llevar la información de forecast
Vendedor de post-venta	Atender al cliente	Darle seguimiento a la venta
		Verificar que el producto llegue a su destino
		Verificar que el producto esté funcionando correctamente
		Elaborar evaluación del producto
		Elaborar reporte de evaluación de venta
	Cambiar mercancía	Elaborar la orden de cambio de producto
		Solicitar revisión de producto al área de soporte
		Solicitar el cambio de producto al encargado de facturación
Vendedor de capacitación y soporte	Administrar recursos	Apartar salas de capacitación
		Llevar la agenda de ingenieros de soporte
	Elaborar plan de trabajo	Hacer plan de trabajo
		Elaborar calendarios de capacitación
		Elaborar listas de precios
	Realizar actividades administrativas	Elaborar orden de facturación
		Solicitar obtener documentos de facturación al asistente
		Solicitar factura al encargado de facturación
	Cotizar	Hacer cotizaciones

		Enviar cotizaciones a clientes
	Reportar ventas	Elaborar reporte de ventas
		Llevar la información de forecast
Administrador de canales	Administrar información distribuidores	Obtener información de distribuidores
		Corroborar la información que se tiene
		Mantener actualizada la base instalada de distribuidores
	Capacitar sobre actualizaciones a distribuidores	Proponer eventos de capacitación técnica para distribuidores
		Proponer eventos de capacitación comercial para distribuidores
		Enviar información y folletos actualizados a distribuidores
		Enviar listas de precios mensualmente
	Apoyar a distribuidores	Visitar a distribuidores
		Visitar clientes como apoyo a distribuidores
Asistente de Gerente de producto	Elaborar documentos	Apoyar en la elaboración de cotizaciones
		Elaborar orden de envío
	Archivar información	Organizar las cotizaciones, órdenes de facturación y envíos
		Almacenar las cotizaciones, órdenes de facturación y envíos
	Controlar pre-facturación	Solicitar documentos a cliente para facturación
		Solicitar comprobante de pago
		Confirmar depósito de pago
		Entregar los documentos a encargado de facturación necesarios para facturar
	Cerrar proceso venta	Confirmar que los productos son entregados
		Archivar la información de la venta
	Ayudar en actividades generales	Atender llamadas
		Enviar e-mail
Recepcionista	Filtrar llamadas	Canalizar las llamadas de los clientes a la persona indicada
	Controlar información de localización	Llevar la agenda de salidas de vendedores
		Mantener actualizado el directorio de empleados
	Enviar productos	Verificar la disponibilidad del mensajero
		Verificar horario de recepción con el cliente
		Entregar croquis de envío al mensajero
Mesajero	Entregar mercancías	Entregar las mercancías en el tiempo asignado
		Llenar reporte de entrega de mercancía
		Entregar reporte de entrega de mercancía al asistente de gerente de producto

Encargado de marketing	Realizar marketing	Obtener folletos de productos
		Obtener productos para marketing
		Hacer telemarketing
	Organizar eventos	Seleccionar el lugar del evento
		Seleccionar la comida
		Elaborar invitaciones
		Elaborar el presupuesto
		Solicitar autorización de evento
		Elaborar evaluaciones del evento
		Entregar evaluaciones del evento
		Enviar invitaciones
		Confirmar asistentes
Encargado de facturación	Facturar	Revisar documentación para facturar
		Realizar factura
		Entregar factura
		Almacenar facturas
	Obtener mercancía	Solicitar producto a encargado de almacén
		Recoger producto de almacén
		Entregar el producto y factura a recepción para su envío
Encargado de almacén	Controlar almacén	Llevar el inventario de almacén
		Acomodar los productos por marca y familia de producto
Encargado de crédito y cobranza	Analizar crédito	Autorizar crédito
		Solicitar documentación para crédito
		Revisar documentación para crédito
		Confirmar al cliente la autorización del crédito
		Confirmar al vendedor la autorización del crédito
	Analizar contabilidad	Revisar el pago de los productos
		Llevar el control de cuentas por cobrar y cuentas por pagar
		Confirmar al asistente de gerente de producto el pago de la compra
Encargado de compras	Administrar información de proveedores	Clasificar el tipo de proveedores
		Llevar la información de proveedores primarios
		Llevar la información de proveedores secundarios
		Mantener actualizada las listas de precios de proveedores
	Controlar pedidos	Elaborar los pedidos
		Solicitar autorización de pedido
		Archivar los pedidos
		Recibir pedidos
		Verificar pedidos

		Entregar pedidos al encargado de almacén

Tabla 1. Tabla de actores, tareas y responsabilidades dentro de la empresa

Con la tabla anterior se obtienen los casos de uso que a continuación se muestran, los cuales nos ayudan a analizar de manera gráfica lo que es importante para la realización del proceso de ventas de manera eficiente de acuerdo a las necesidades de la empresa, de esta manera se puede definir los límites del sistema para la mejora de este proceso, así como las personas que lo utilizan y los elementos necesarios para su correcto funcionamiento.

El caso de uso de la figura VI.1 nos muestra el sistema de ventas de manera global en el cual se pueden identificar todos los actores que participan y las principales tareas que realizan para el proceso de ventas.

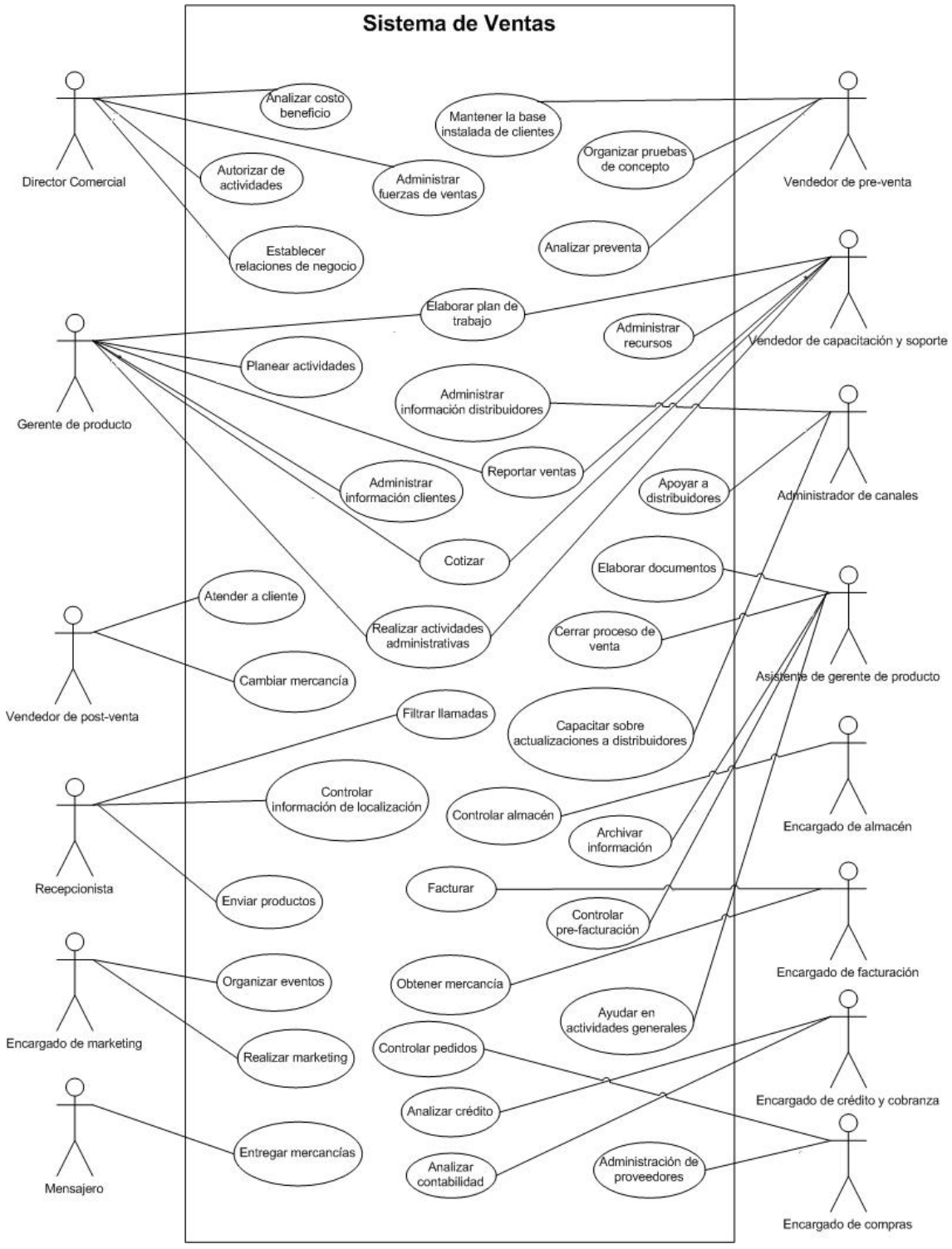


Figura VI.1. Caso de uso para el sistema global de ventas

Las figuras que a continuación se muestran que abarcan de la figura VI.2 a la figura VI.41, contienen los casos de uso desglosados por actor, en base a las responsabilidades que cada tarea conlleva.

Esto nos ayuda a tener una descomposición por niveles en el que se pueden identificar las responsabilidades mayores, que conlleva realizar cada una de las tareas mencionadas anteriormente.

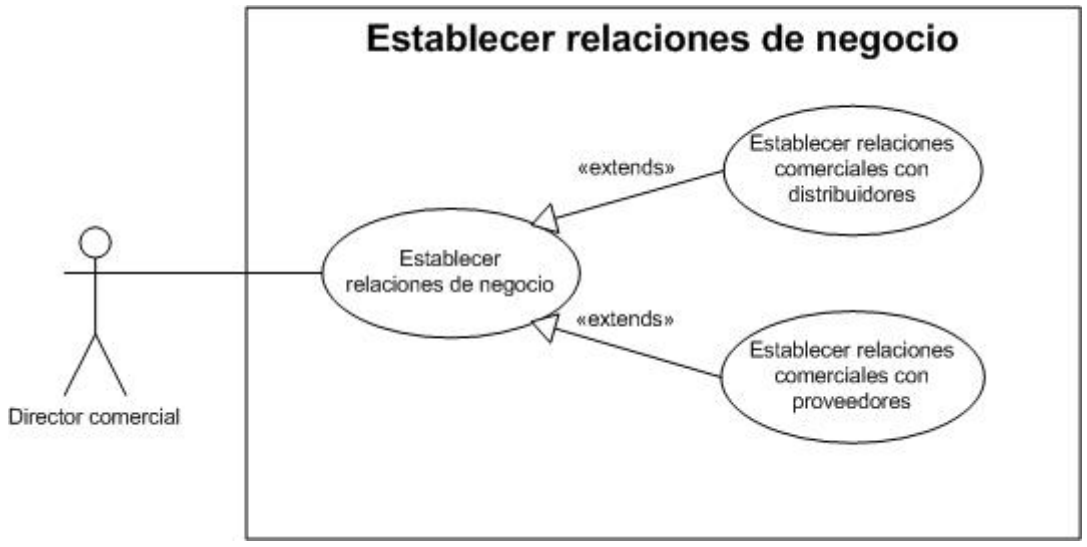


Figura VI.2. Caso de uso para establecer relaciones de negocio



Figura VI.3. Caso de uso para administrar fuerzas de ventas

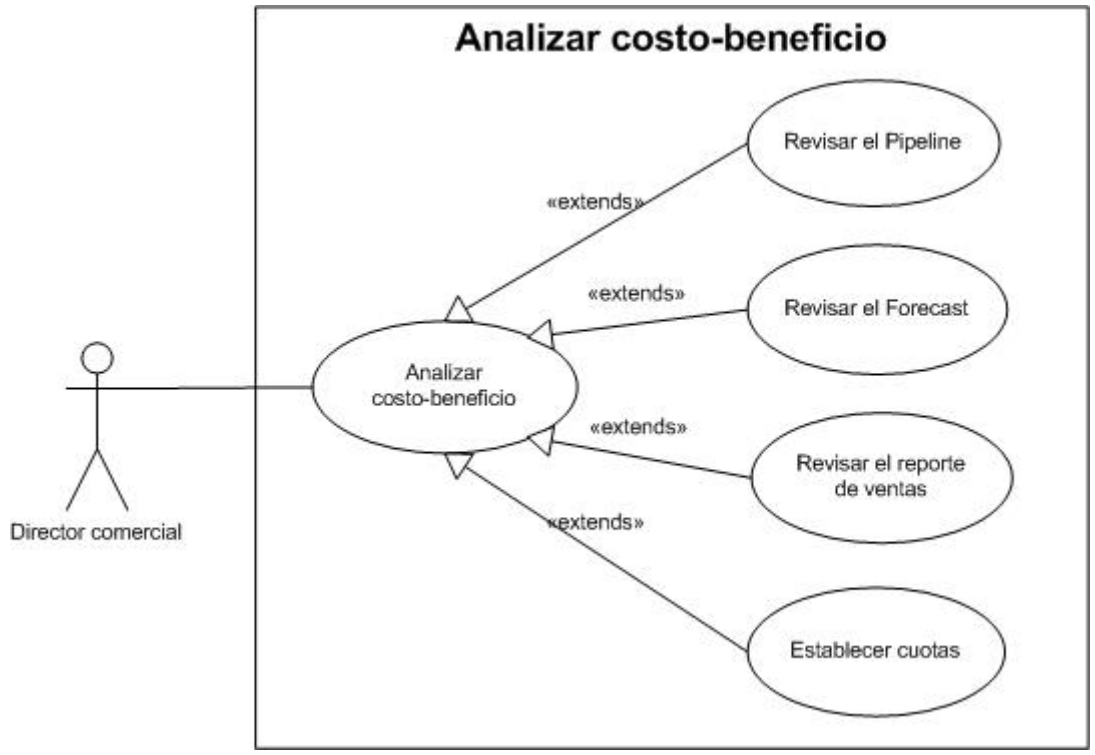


Figura VI.4. Caso de uso para analizar costo-beneficio

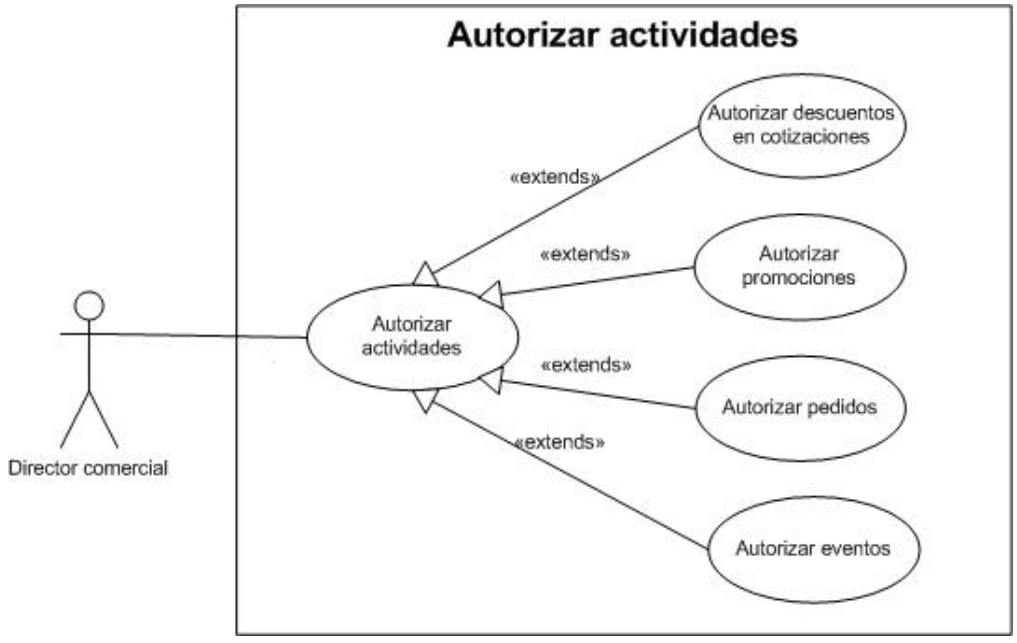


Figura VI.5. Caso de uso para autorizar actividades

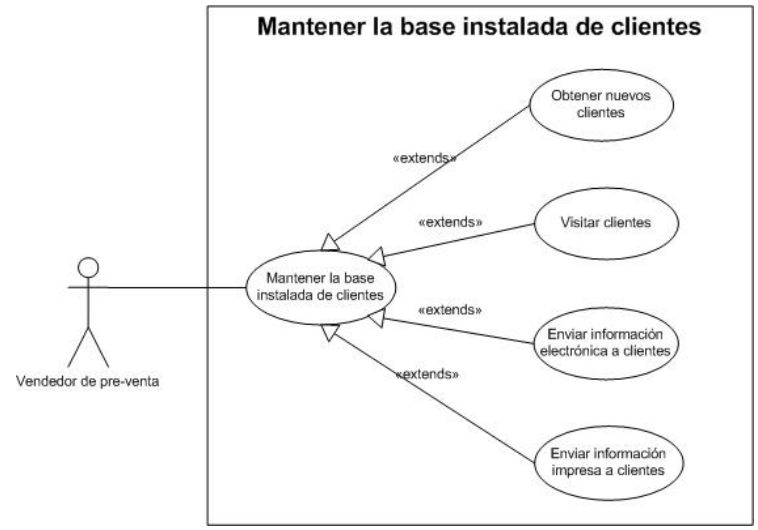


Figura VI.6. Caso de uso para organizar pruebas de concepto

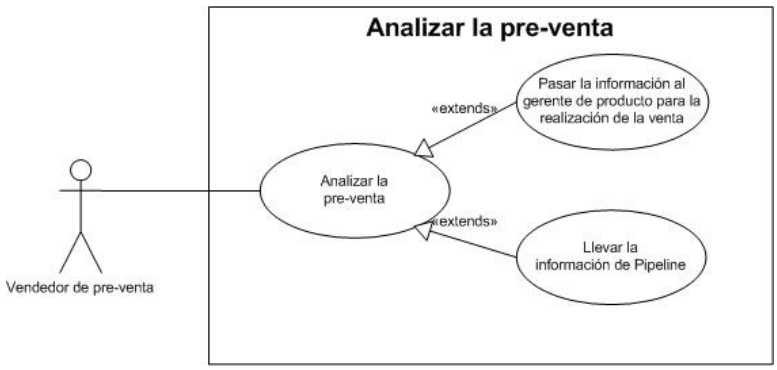


Figura VI.7. Caso de uso para analizar la pre-venta

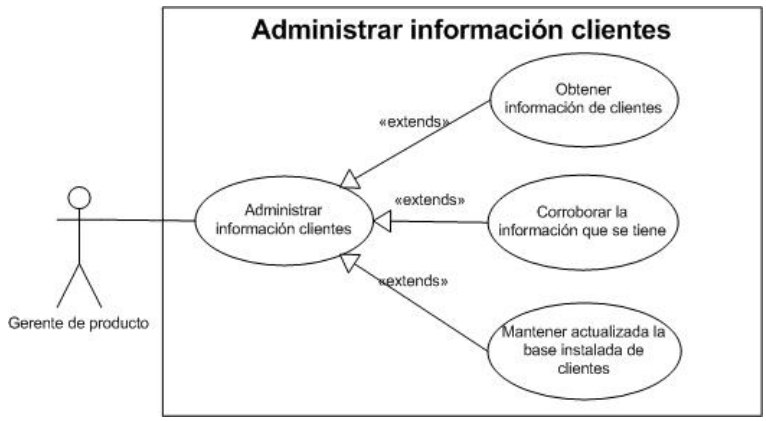


Figura VI.8. Caso de uso para administrar información de clientes

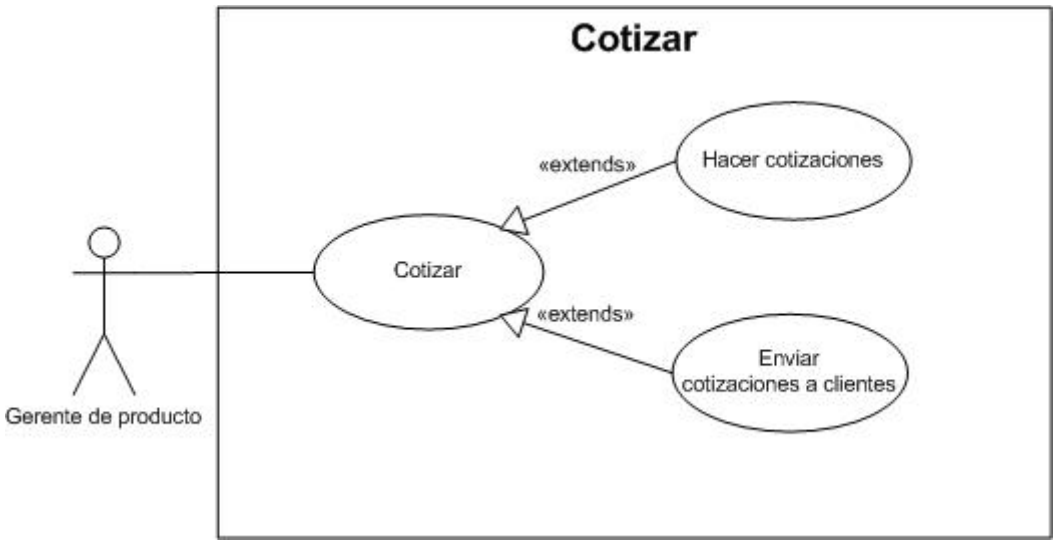


Figura VI.9. Caso de uso para cotizar

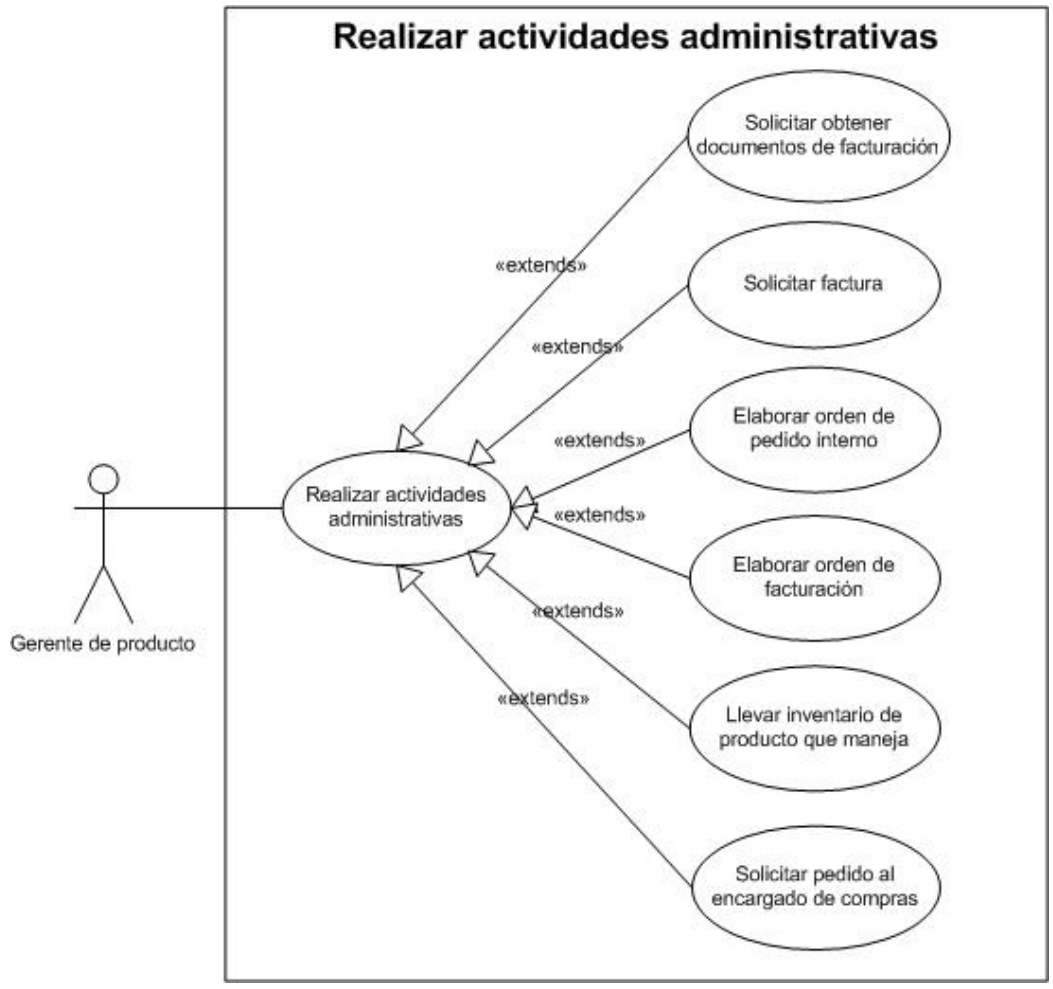


Figura VI.10. Caso de uso para realizar actividades administrativas

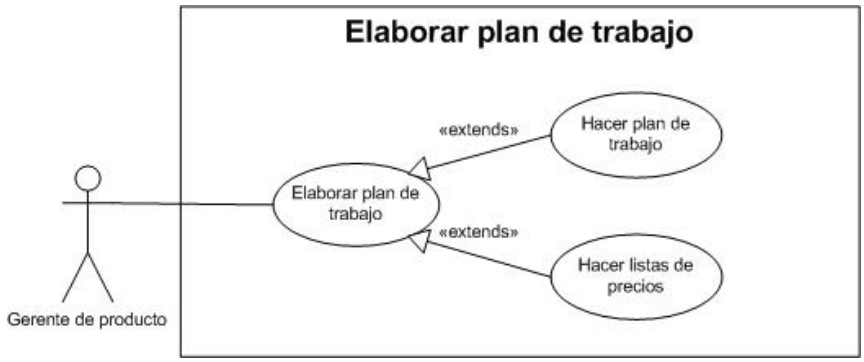


Figura VI.11. Caso de uso para elaborar plan de trabajo

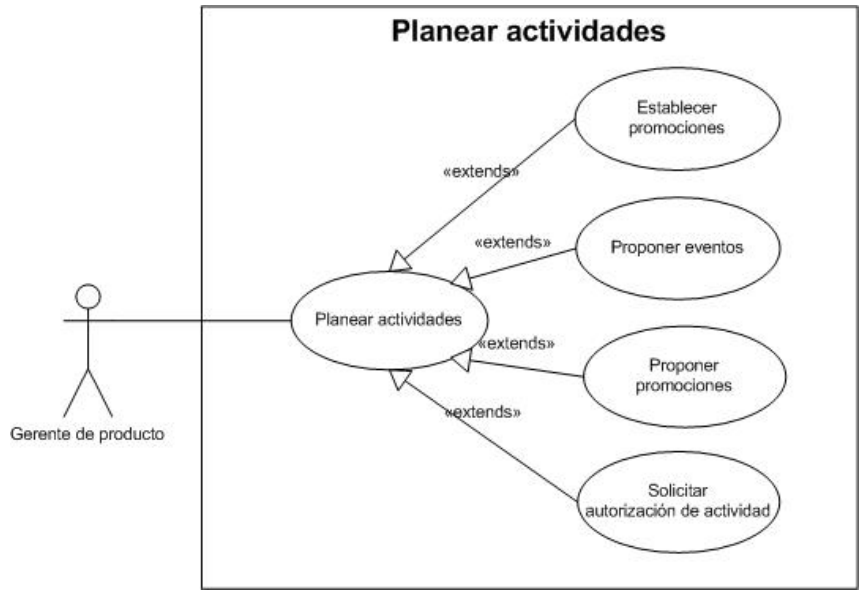


Figura VI.12. Caso de uso para planear actividades

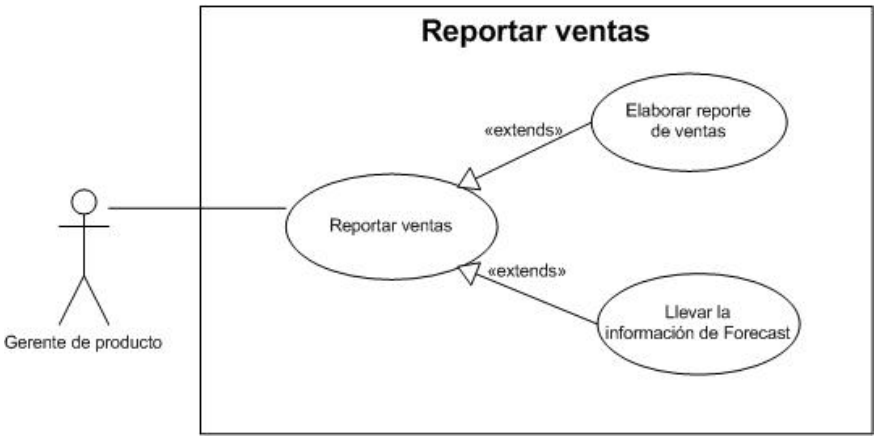


Figura VI.13. Caso de uso para reportar ventas

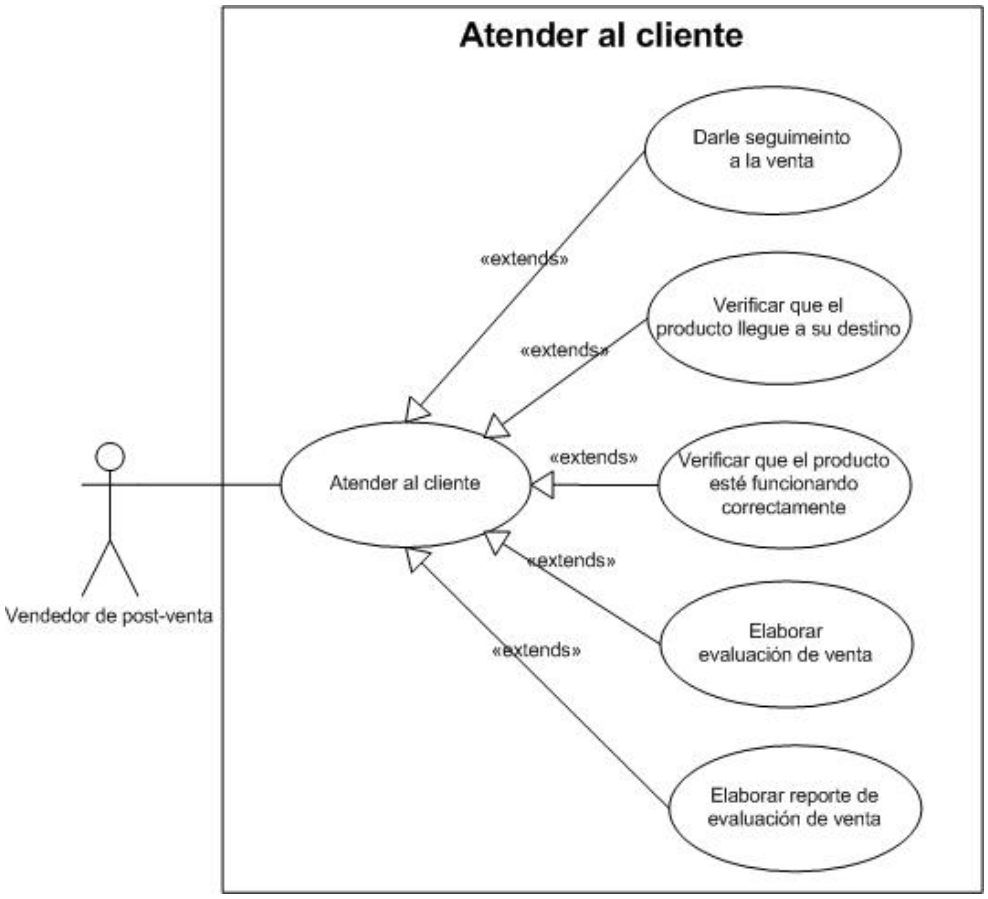


Figura VI.14. Caso de uso para atender al cliente

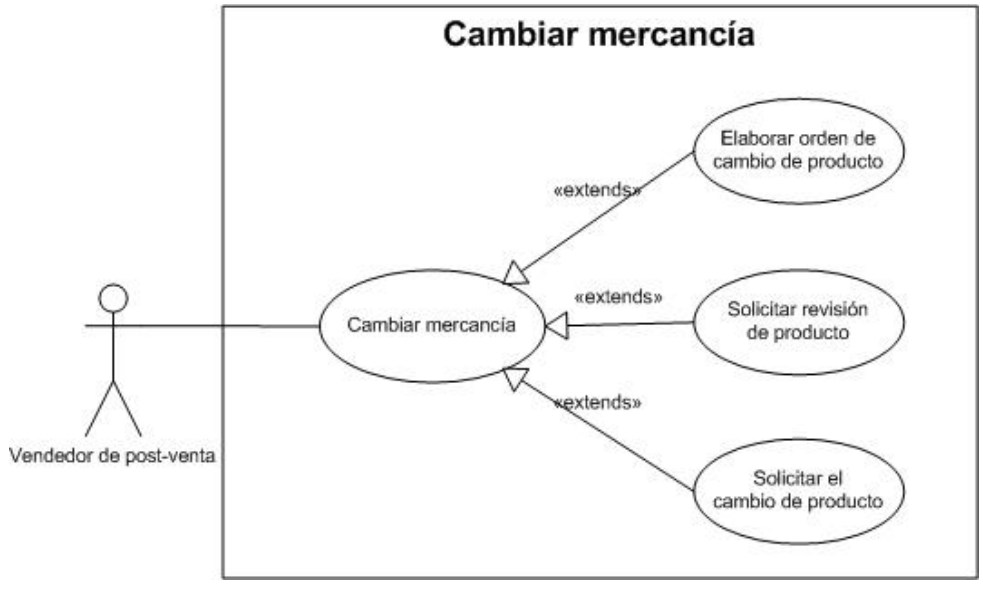


Figura VI.15. Caso de uso para cambiar mercancía

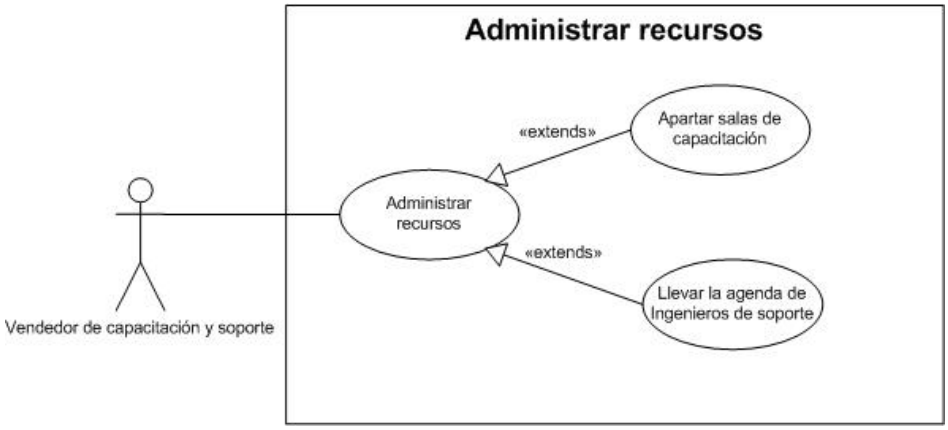


Figura VI.16. Caso de uso para administrar recursos

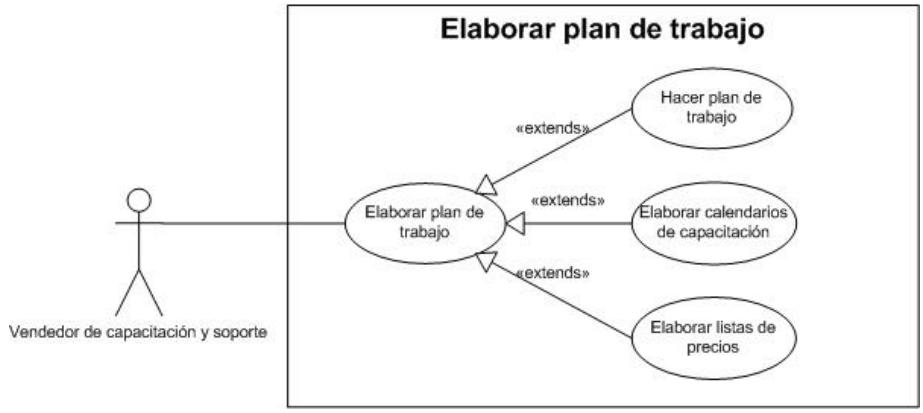


Figura VI.17. Caso de uso para elaborar plan de trabajo

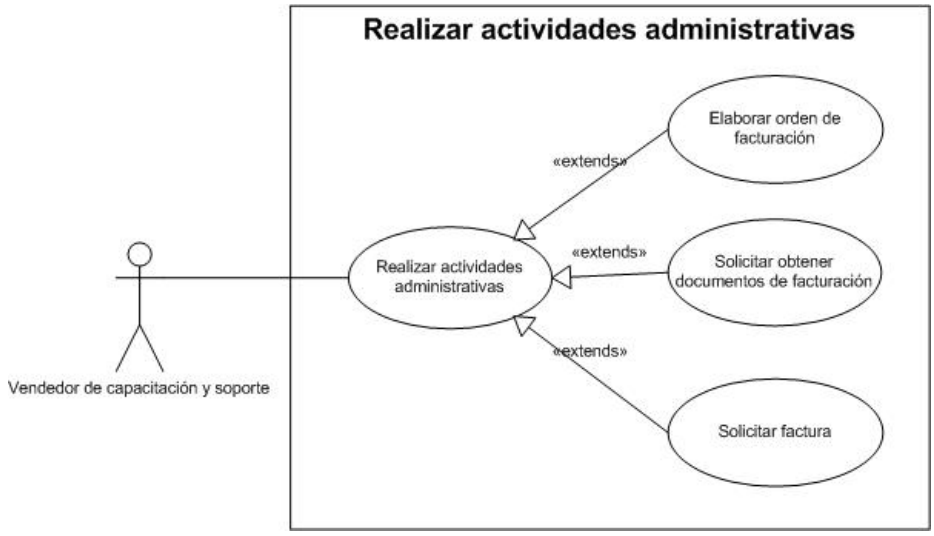


Figura VI.18. Caso de uso para realizar actividades administrativas

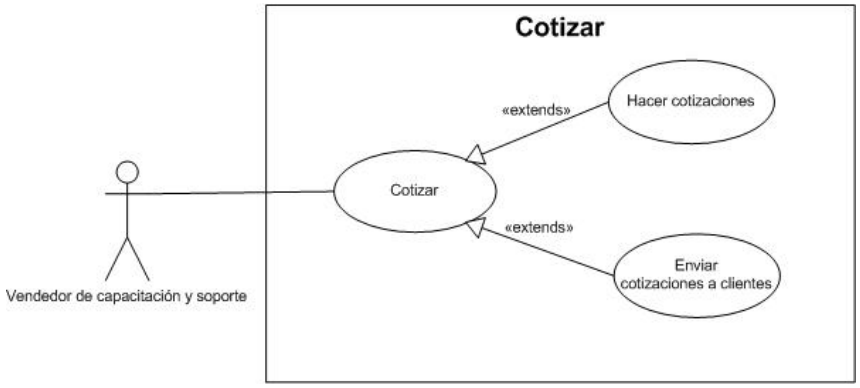


Figura VI.19. Caso de uso para cotizar

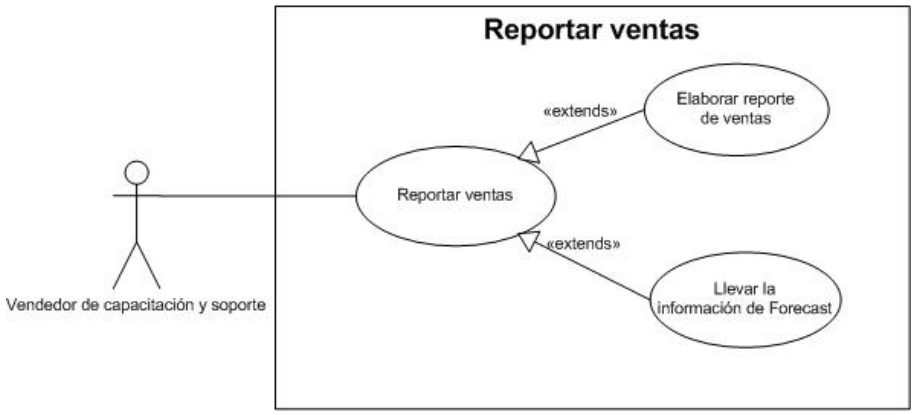


Figura VI.20. Caso de uso para reportar ventas

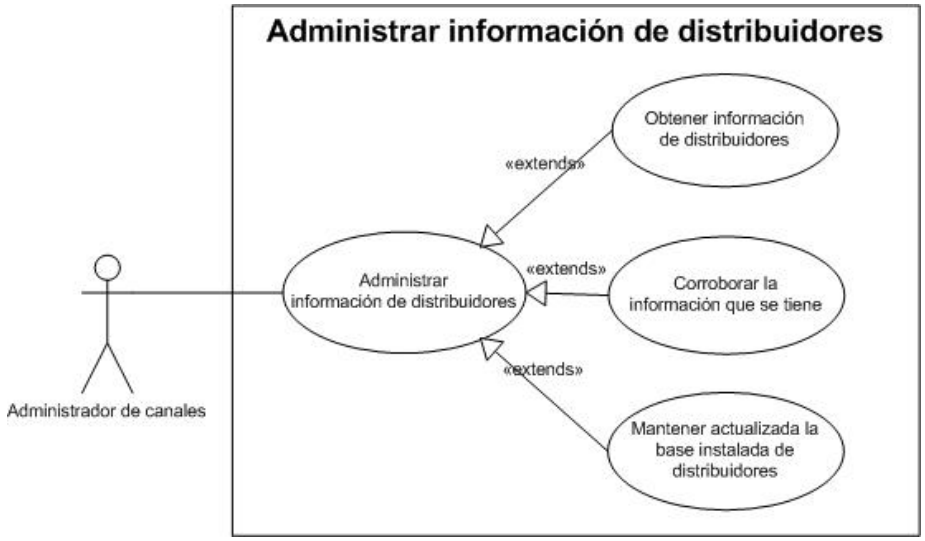


Figura VI.21. Caso de uso para administrar información de distribuidores

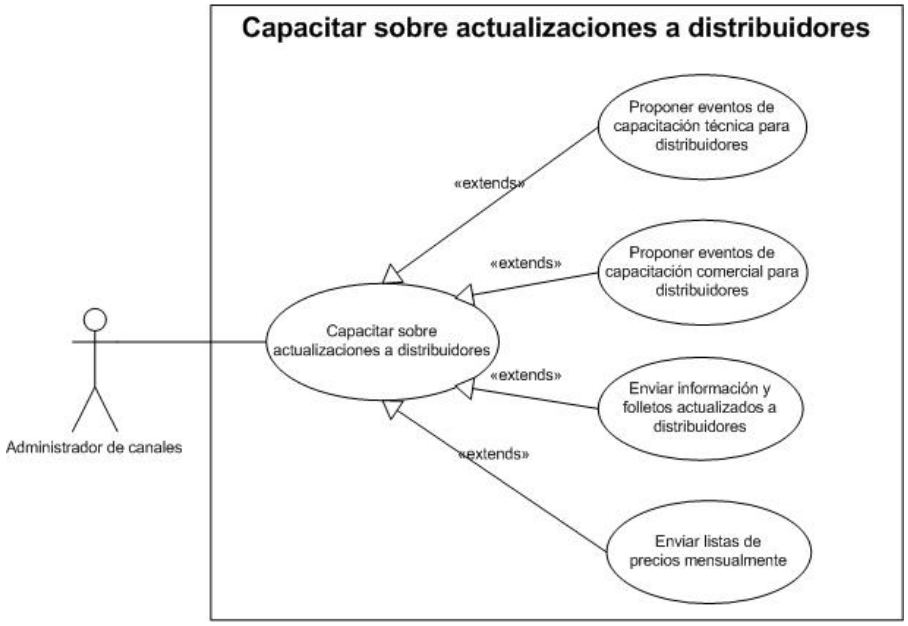


Figura VI.22. Caso de uso para capacitar sobre actualizaciones a distribuidores

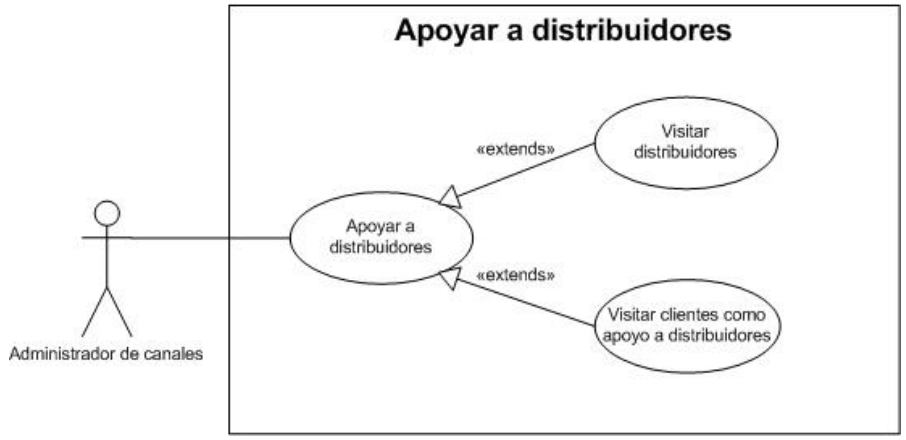


Figura VI.23. Caso de uso para apoyar a distribuidores

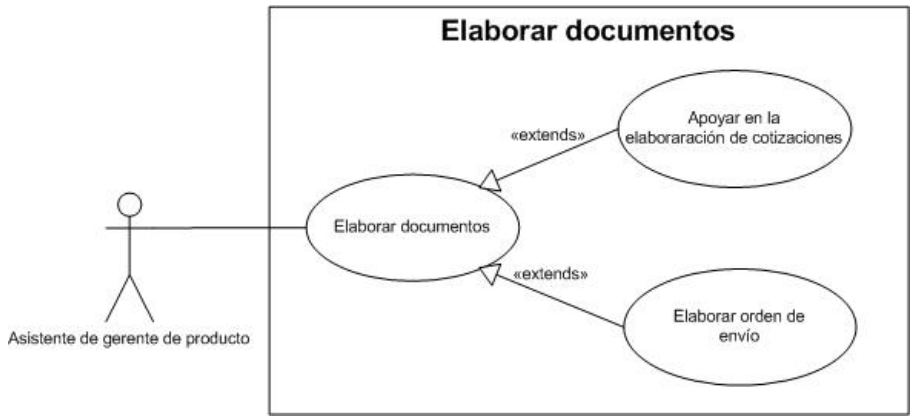


Figura VI.24. Caso de uso para elaborar documentos

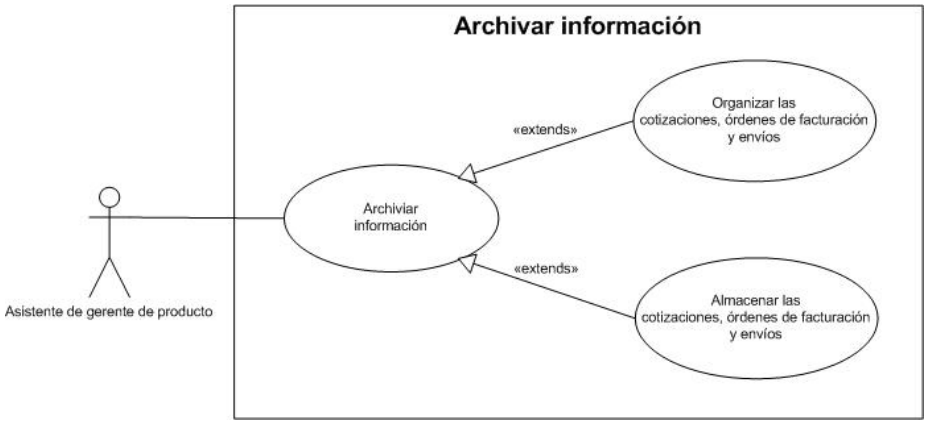


Figura VI.25. Caso de uso para archivar información

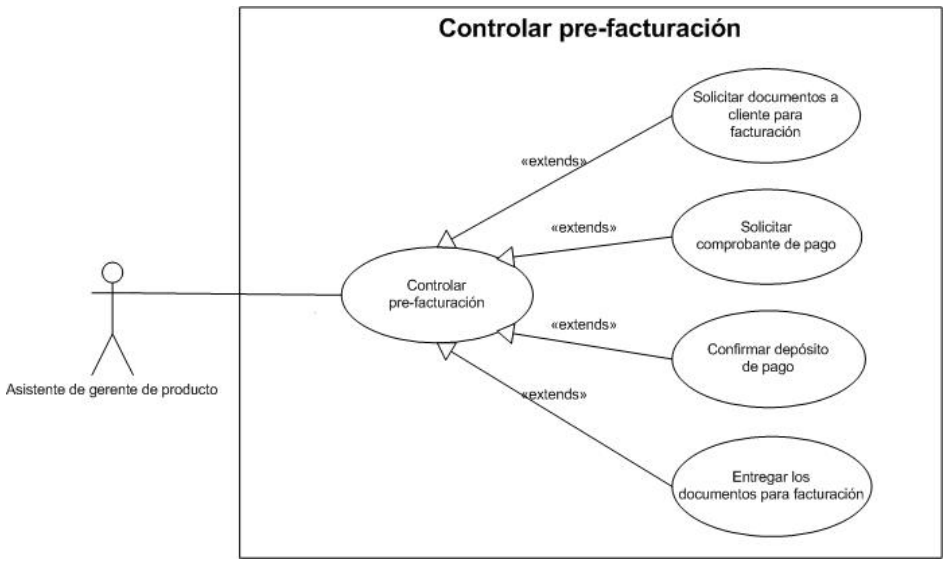


Figura VI.26. Caso de uso para controlar pre-facturación

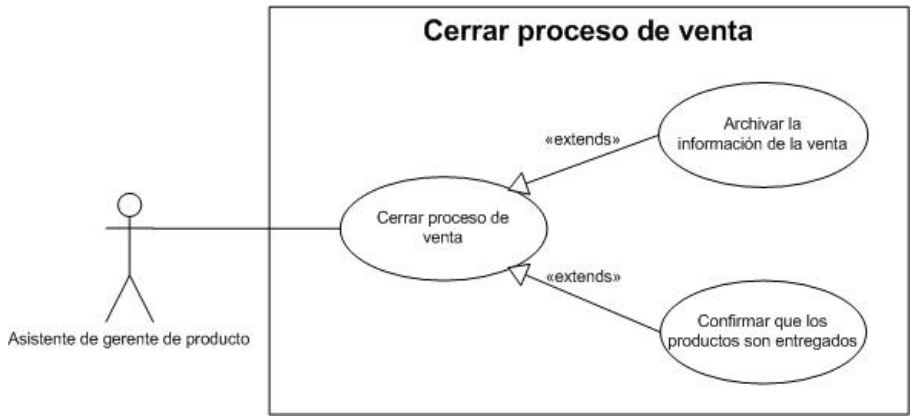


Figura VI.27. Caso de uso para cerrar proceso de venta

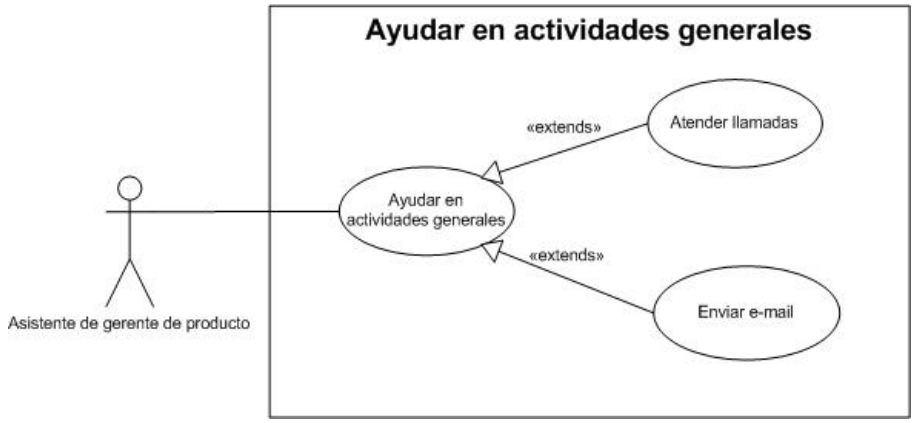


Figura VI.28. Caso de uso para ayudar en actividades generales

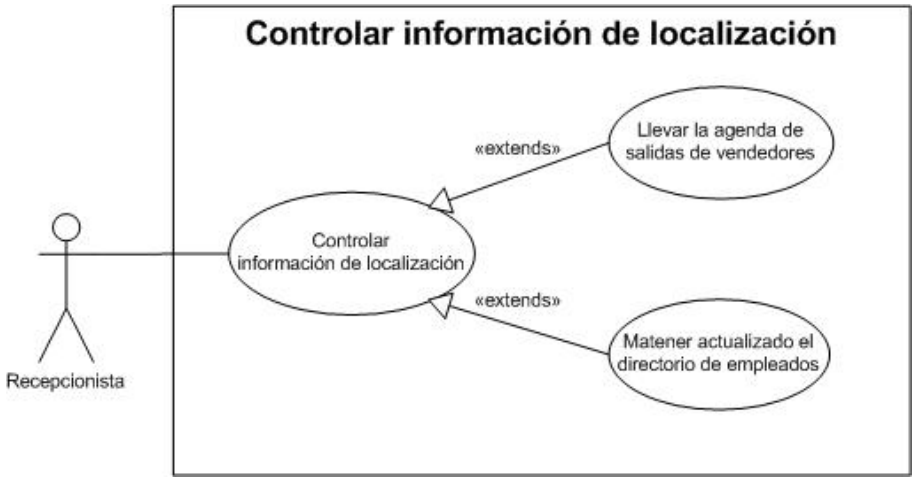


Figura VI.29. Caso de uso para controlar información de localización

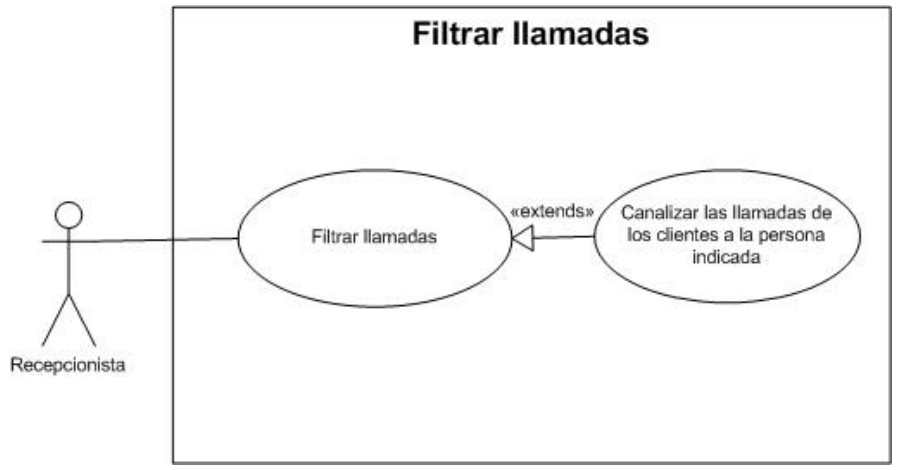


Figura VI.30. Caso de uso para filtrar llamadas

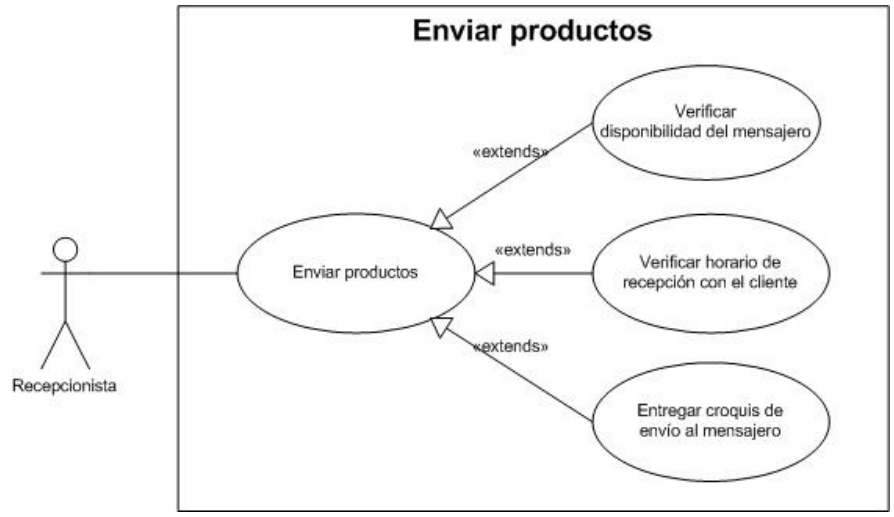


Figura VI.31. Caso de uso para enviar productos

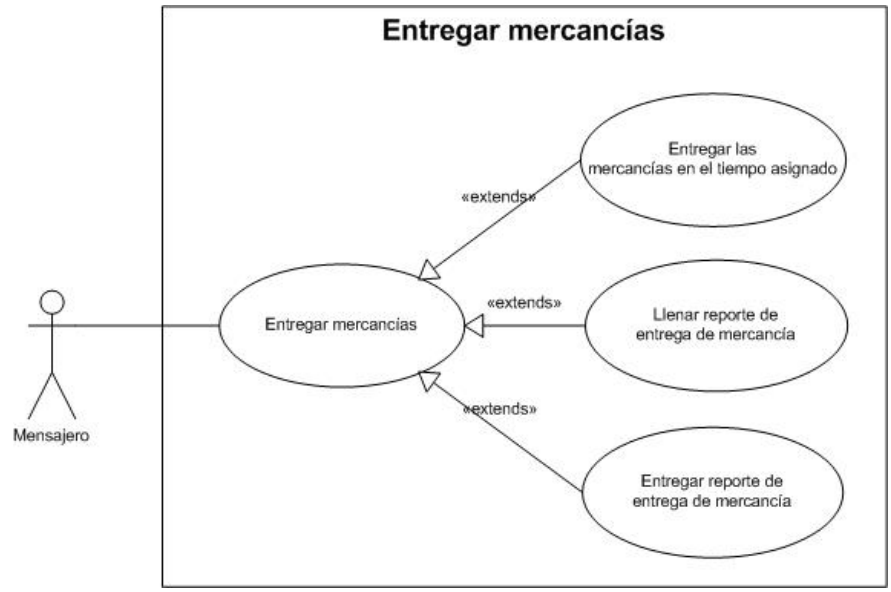


Figura VI.32. Caso de uso para entregar mercancías

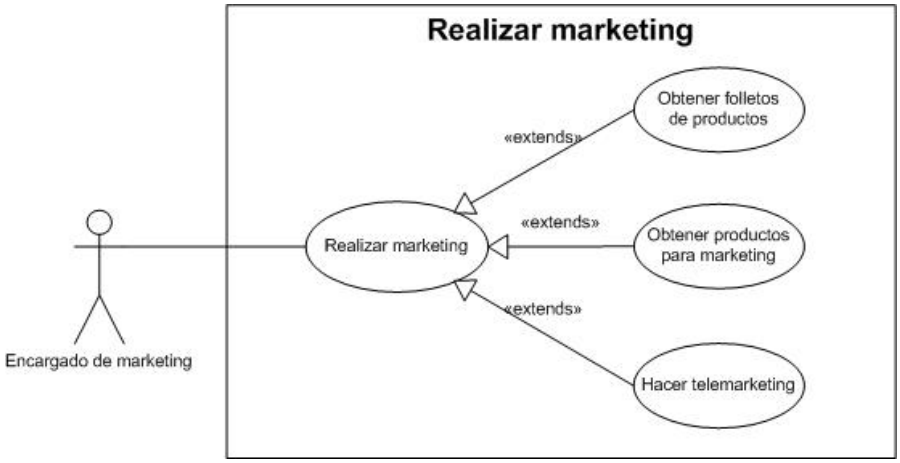


Figura VI.33. Caso de uso para realizar marketing

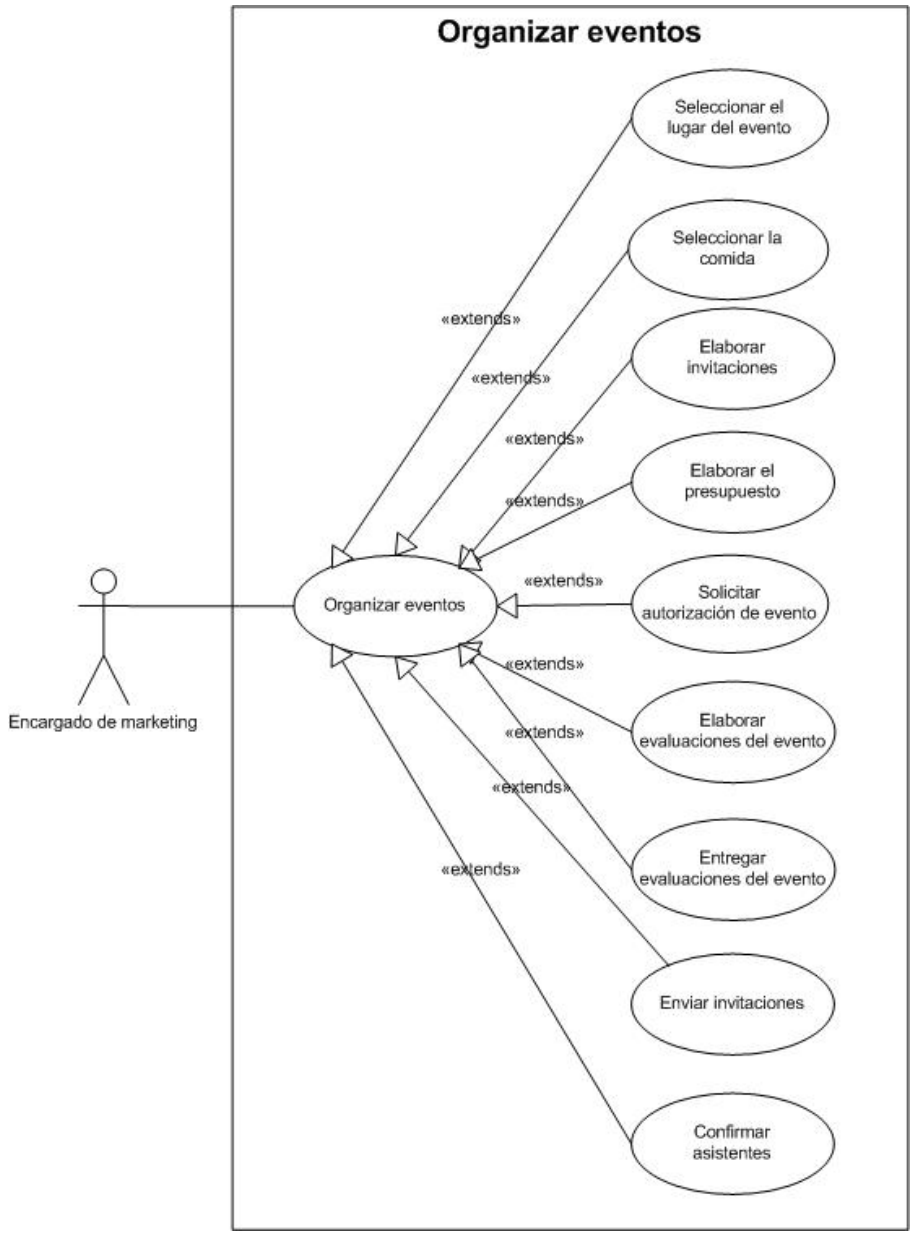


Figura VI.34. Caso de uso para organizar eventos

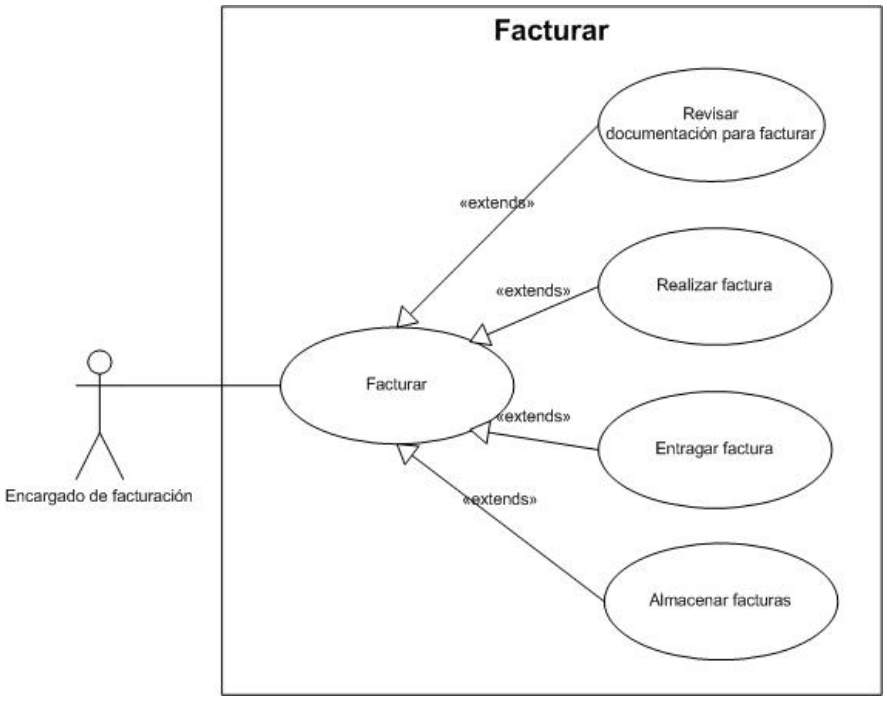


Figura VI.35. Caso de uso para facturar

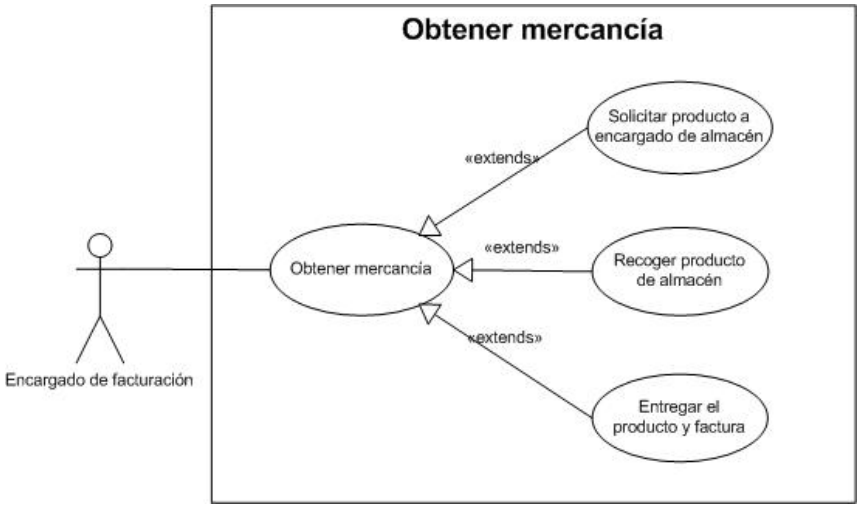


Figura VI.36. Caso de uso para obtener mercancía

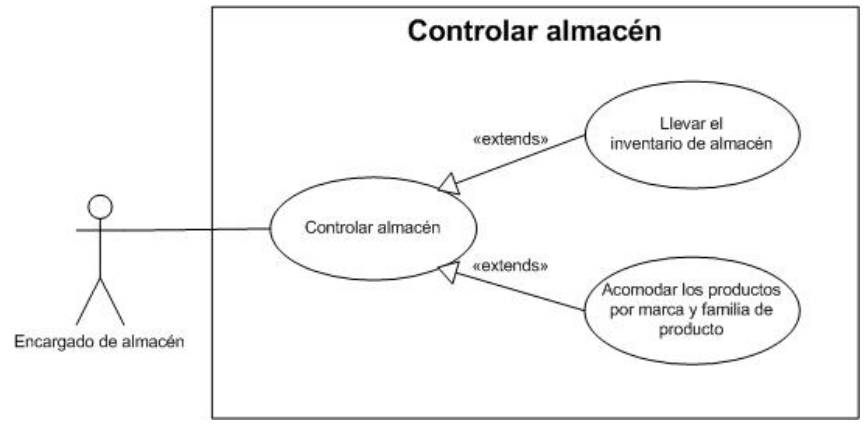


Figura VI.37. Caso de uso para controlar almacén

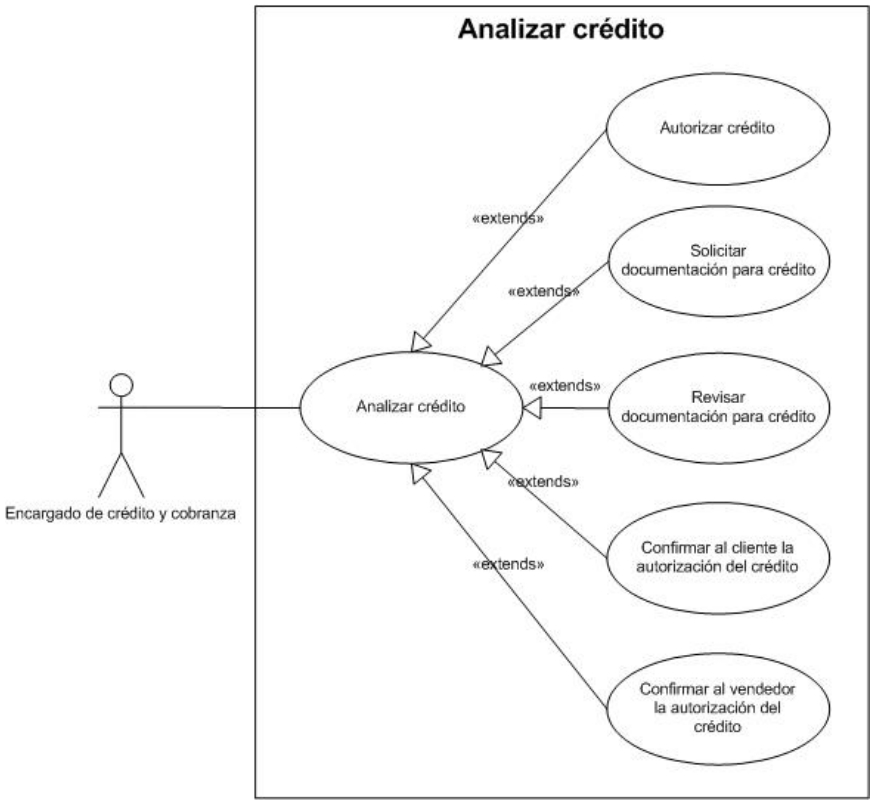


Figura VI.38. Caso de uso para analizar crédito

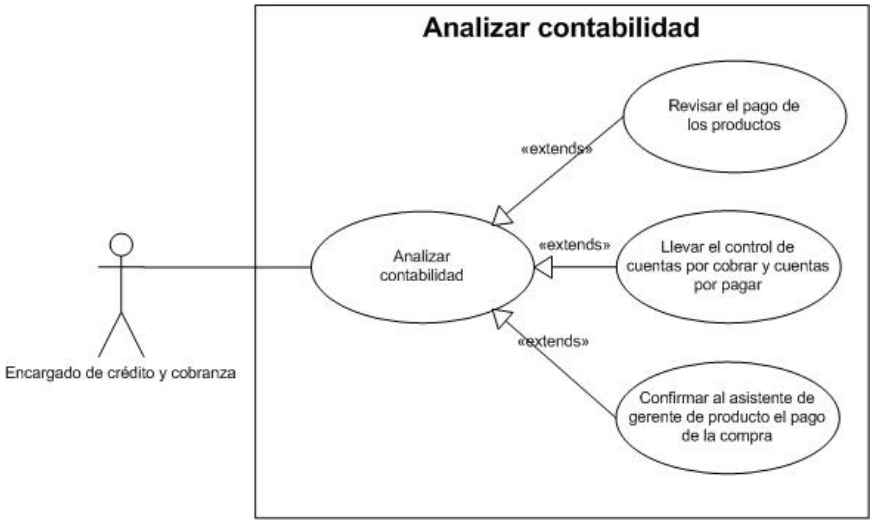


Figura VI.39. Caso de uso para analizar contabilidad

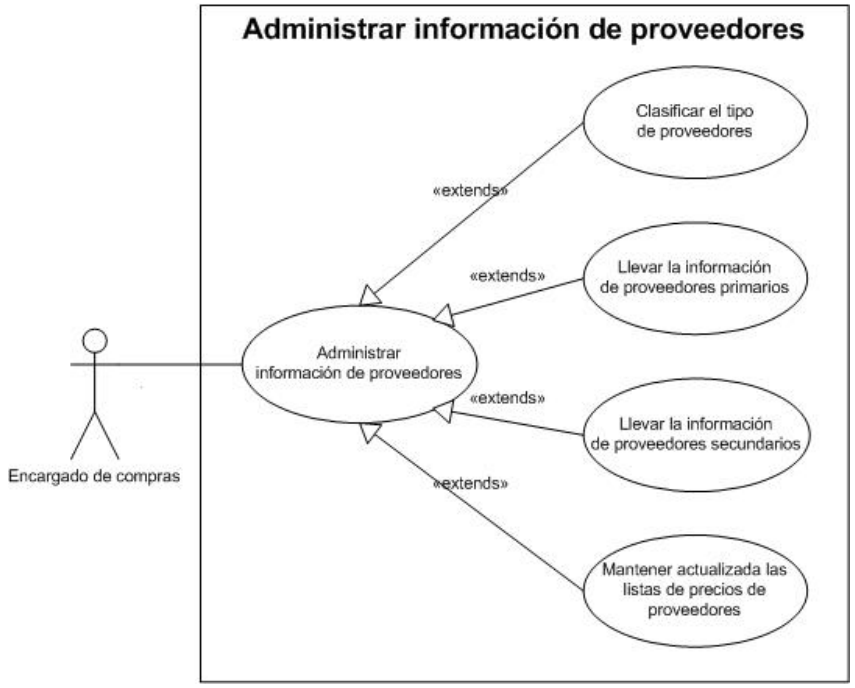


Figura 40. Caso de uso para administrar información de proveedores

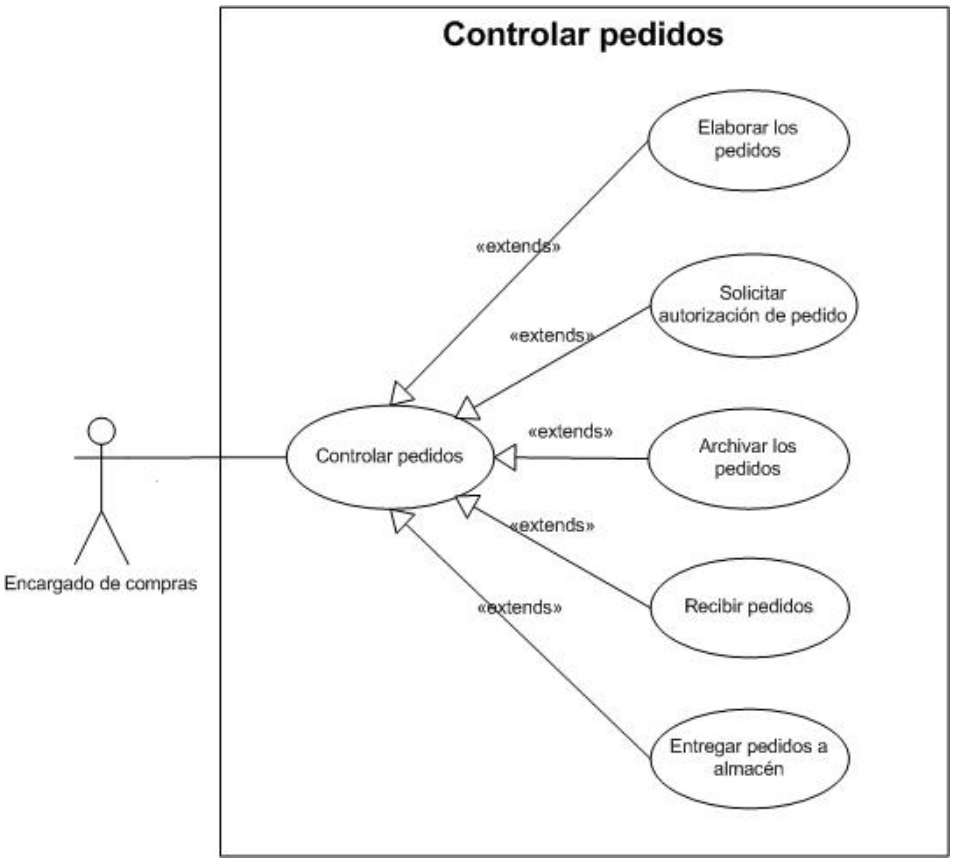


Figura VI.41. Caso de uso para controlar pedidos

En base a los casos de uso mostrados en las figuras anteriores, se puede realizar el análisis del proceso de ventas adecuado para la empresa. Para que posteriormente, una vez que se tenga el proceso establecido pueda analizarse a través de diagramas de secuencia las iteraciones entre objetos de manera cronológica.

A continuación se muestra la descripción del proceso de ventas, para poder hacer posteriormente el análisis de cómo se realiza el proceso a través de diagramas de secuencia.

- Analizar del tipo de cliente: Proceso en el que se identifica el tipo de cliente para tener la información necesaria para hacer una cotización.
  - Usuario Final
    - Se identifica el tipo de usuario final
      - Cliente registrado
        - Se verifica que los datos que se tienen sean los correctos
      - Nuevo cliente
        - Se registran los datos

- Distribuidor
  - Se identifica el tipo de distribuidor
    - Distribuidor dado de alta
      - Identificar que clasificación de distribuidor tiene para conocer el tipo de descuento que se puede ofrecer
        - Especial
        - Certificado
        - Registrado
    - Distribuidor no dado de alta
      - Se le envía el formato de Alta de distribuidor, y se le solicitan los documentos necesarios para darlo de alta como son:
        - Comprobante de domicilio
        - Comprobante de Alta de Hacienda
        - Copia de RFC

- Pre-venta: Proceso en el que se realiza el seguimiento necesario antes del proceso de venta.
  - Seguimiento del proyecto de compra
    - Llamadas telefónicas
      - Explicación breve del producto, aclaración de dudas y comentarios
    - Demostración del producto
      - Seminarios
      - Presentaciones
    - Envío de información
      - Folletos en formato electrónico enviados por e-mail
      - Folletos impresos enviados por mensajería o correo
    - Pruebas de evaluación del producto (todo este proceso se debe hacer en conjunto con el área de soporte por lo que es muy importante organizar los tiempos de los Ingenieros de soporte para la realización de las pruebas y la elaboración de los documentos en conjunto).
      - Elaboración de la prueba de concepto

- Se establece el lugar en el que se van a realizar las pruebas de evaluación
    - Instalaciones del cliente
    - Instalaciones de la empresa
- Se llena el documento de prueba de concepto en la que se verifica junto con el cliente y el Ingeniero de soporte que se cuenta con el equipo adecuado e infraestructura mínima y necesaria para la realización de las pruebas
- Se establece la situación actual antes de la realización de las pruebas y qué se va a hacer dentro de la prueba, para qué se va a realizar, por qué y quienes la llevarán a cabo
- Una vez que se establecen las bases necesarias y el cliente está de acuerdo, el documento de prueba de concepto se firma por ambas partes (el cliente y la empresa)
- Se realiza la prueba

♦ Elaborar reporte de pre-venta

o Entregar reporte de pre-venta al Gerente de producto para su la realización de la venta
o Elaborar el reporte de pipeline para el Director comercial

❖ Cotizaciones: Proceso en el que se realiza la cotización del producto que ha sido solicitado, incluyendo el descuento si es que aplica.

➢ Identificar el producto o productos a ser cotizados en base a los requerimientos del cliente
➢ Confirmar que lo que se va a cotizar es o son los productos que el cliente necesita
➢ Elaborar la cotización con la información necesaria

♦ No. de cotización
♦ Cantidad
♦ Descripción breve del producto
♦ Precio unitario
♦ IVA
♦ Descuento (si aplica)
♦ Total
♦ Condiciones comerciales

o Tipo de cambio
o Tipo de pago
o Tiempo de entrega
o Validez

➢ Se archiva la cotización

- Venta: Proceso en el que se realiza la venta. Una vez que se tiene la orden de compra y el pago de la misma se procede al proceso de Facturación.

  - Se obtiene la orden de compra firmada por el cliente
  - Se verifica que se cuenta en almacén con el producto solicitado, en caso contrario se procede al proceso de pedido interno de producto.
  - Se solicita el envío del comprobante de pago

    - Envío por e-mail o Fax

  - Se verifica en el área administrativa con el comprobante impreso, que se cuenta con el depósito correspondiente
  - Se procede al proceso de facturación
  - Facturación se encarga de solicitar los productos y del envío correspondiente
  - Una vez que se confirma la entrega del producto al recibir el informe del envío por parte del mensajero se almacena la información de la venta

- Post-venta: Proceso en el que se da seguimiento a la venta realizada. Una vez que el proceso de venta procede al de facturación el proceso de post-venta empieza su tarea.

  - Atención al cliente

    - Darle seguimiento a la venta
    - Recabar evaluaciones del producto
    - Verificar que el producto funciona correctamente

  - Elaborar reporte de evaluación de venta

  - Cambio de mercancía

    - Verificar con el área de soporte que el producto no funcione correctamente
    - Elaborar orden de cambio de producto
    - Solicitar cambio de producto
    - Entregar el nuevo producto al cliente

- Pedido interno de productos: Proceso por medio del cual el vendedor hace un pedido interno al área de compras, para que éste lo solicite a los proveedores correspondientes.

  - Cada vez que se realiza una venta el inventario de productos en almacén tiene una variación por lo que cuando algún producto está por tener existencia nula en almacén, el vendedor hace un nuevo pedido al encargado de compras.
  - Se elabora la orden de pedido interno con la información de:

- No. de pedido
- Nombre del producto
- Cantidad
- Precio unitario
- Descuento
- Total
- Nombre del proveedor
- Firma de autorización del pedido por parte del vendedor

- Se solicita el producto al encargado de compras
- Se procede al proceso de compras

- Facturación: Proceso que se lleva a cabo para la realización de la factura correspondiente a la venta.

  - Elaboración de orden de facturación, con los datos correspondientes.

    - No. de orden de facturación
    - Fecha
    - Nombre del cliente
    - Descripción del producto
    - Clave del producto
    - Precio
    - Descuento (si aplica)
    - IVA
    - Tipo de cambio (si la factura es en M. N.)
    - Total

  - Si es la primera vez que se va a facturar a un cliente es necesario solicitarle unos documentos adicionales para poder darlo de alta en el sistema de facturación. En el caso de los distribuidores, ya se debe de contar con estos documentos puesto que es un requisito para registrarlos.

    - Comprobante de alta de hacienda
    - Comprobante de domicilio
    - Copia de RFC

  - Se entrega la orden de facturación al área administrativa, para la realización de la factura correspondiente.
  - El encargado de facturación solicita los productos correspondientes al área de almacén
  - Una vez que almacén entrega los productos solicitados al encargado de facturación, este realiza la factura con la información que se cuenta en el sistema o se inserta la información proporcionada por el área de ventas
  - Entrega la factura junto con los productos y la orden de envío de productos a recepción para la planeación del envío de productos

- Envío de productos: Proceso que se lleva a cabo para el envío de los productos a las instalaciones del cliente.

  - México D.F. y Área Metropolitana.

- En caso de que el producto se envía dentro de esta área, se entrega la orden de envío junto con la orden de facturación para que se programe el envío a través de los mensajeros de la empresa.
- Se solicita a la empresa confirmar hora de recepción de mercancía para programar el envío
- Se verifica la disponibilidad del mensajero
- Se programa el envío
- Se entrega el producto
- Se obtiene la firma de recepción del producto
- Se proporciona el reporte de envío al vendedor indicado

- Provincia

  - En caso de enviar el producto a provincia, se debe entregar esta la orden de envío junto con la orden de facturación para la programación del envío oportuno por medio de mensajería.

- Autorización de crédito: Proceso que se lleva a cabo cuando el cliente pide crédito para el pago de los productos.

  - Se solicita la documentación para la evaluación de la autorización del crédito
  - Se analiza la documentación
  - Se autoriza el crédito si se cumplen con los requisitos
  - Se confirma al vendedor y al cliente la autorización del crédito

- Compras: Proceso que se lleva a cabo cuando el encargado de compras recibe la solicitud de compra de producto por el área de ventas.

  - Se recibe la solicitud de pedido interno por parte del vendedor
  - Se busca información por parte de los proveedores para la elaboración del pedido de productos
  - Se elabora el pedido para su autorización
  - Se solicita autorización de pedido al Director comercial
  - Una vez que ha sido autorizado se hace el pedido al proveedor correspondiente
  - Una vez que se obtiene la mercancía se entrega al área de almacén
  - El almacén actualiza su inventario
  - El almacén entrega el reporte actual del inventario al vendedor

- Marketing: Proceso que se lleva a cabo cuando el vendedor necesita ayuda de mercadotecnia para la promoción de sus productos.

  - Se planean los eventos

    - Selección del lugar
    - Selección de la comida
    - Selección de la mercadotecnia a usarse

- Productos de marketing
- Folletos de productos

- ♦ Selección de invitaciones
- ♦ Elaboración de invitaciones
- ♦ Envío de invitaciones
- ♦ Confirmación de asistencia a eventos
- ♦ Realización de telemarketing

- ➢ Se elabora el presupuesto económico para la realización de eventos
- ➢ Se solicita autorización de presupuesto para la realización del evento
- ➢ Se realiza el evento
- ➢ Se elaboran las evaluaciones para los eventos que se realizan, para obtener información de los clientes de interés de los vendedores
- ➢ Se entregan los reportes de las evaluaciones

- ❖ Administración de recursos: Proceso que se lleva a cabo cuando el vendedor necesita algún recurso adicional del área de soporte técnico como apoyo a su venta o como parte de su venta.

  - ➢ Capacitación

    - ♦ Se verifica la disponibilidad de salas para capacitación
    - ♦ Se verifica la disponibilidad del área de soporte técnico para impartir la capacitación
    - ♦ Se agenda la capacitación a la persona responsable
    - ♦ Se confirma la capacitación

  - ➢ Soporte técnico

    - ♦ Se verifica la disponibilidad de los Ingenieros de soporte.
    - ♦ Se analiza la forma de brindar soporte técnico
    - ♦ Se agenda la actividad con la persona indicada
    - ♦ Se confirma la actividad
    - ♦ Se realiza la actividad
    - ♦ Se elabora el reporte de actividad como parte del historial de soportes.

En base al proceso propuesto, se tienen los siguientes diagramas de secuencia mostrados de la figura VI.42 a la figura VI. 59, para que con ayuda de estos se pueda analizar de manera cronológica la forma en que se desarrolla el proceso, así como los mensajes entre los objetos que lo forman.

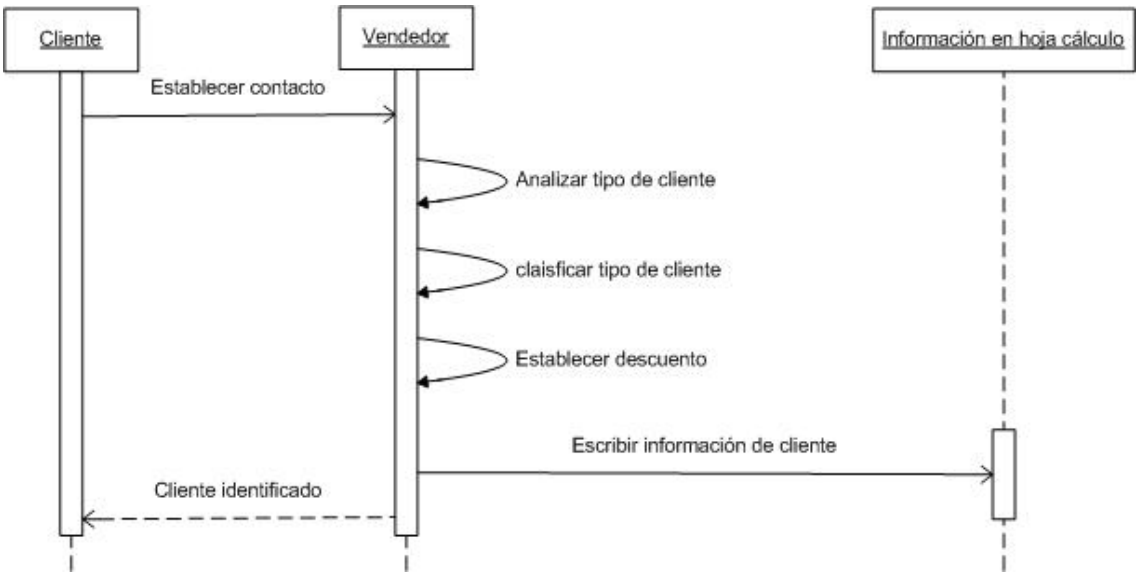


Figura VI.42. Diagrama de secuencia para el módulo de análisis de tipo de cliente

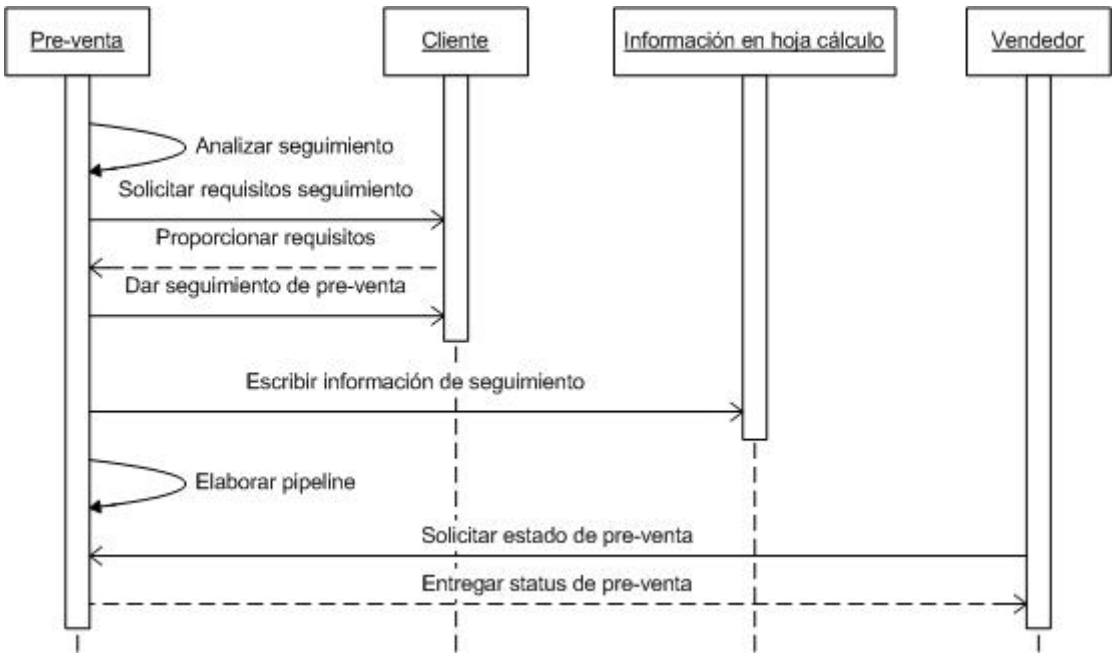


Figura VI.43. Diagrama de secuencia para el módulo de seguimiento de pre-venta

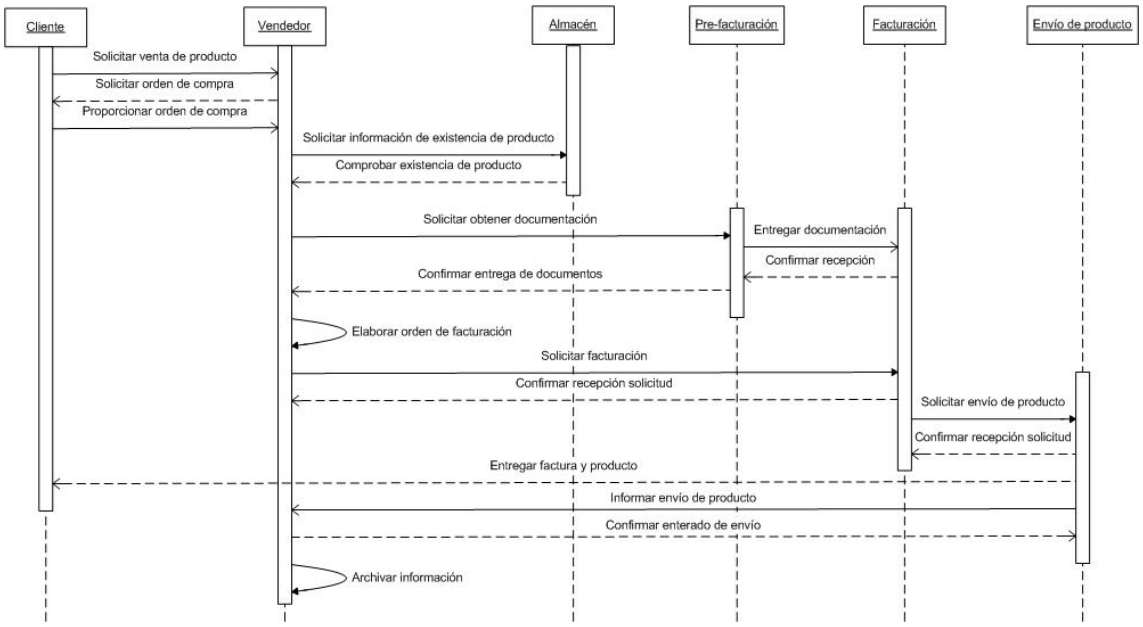


Figura VI.44. Diagrama de secuencia para el módulo de venta

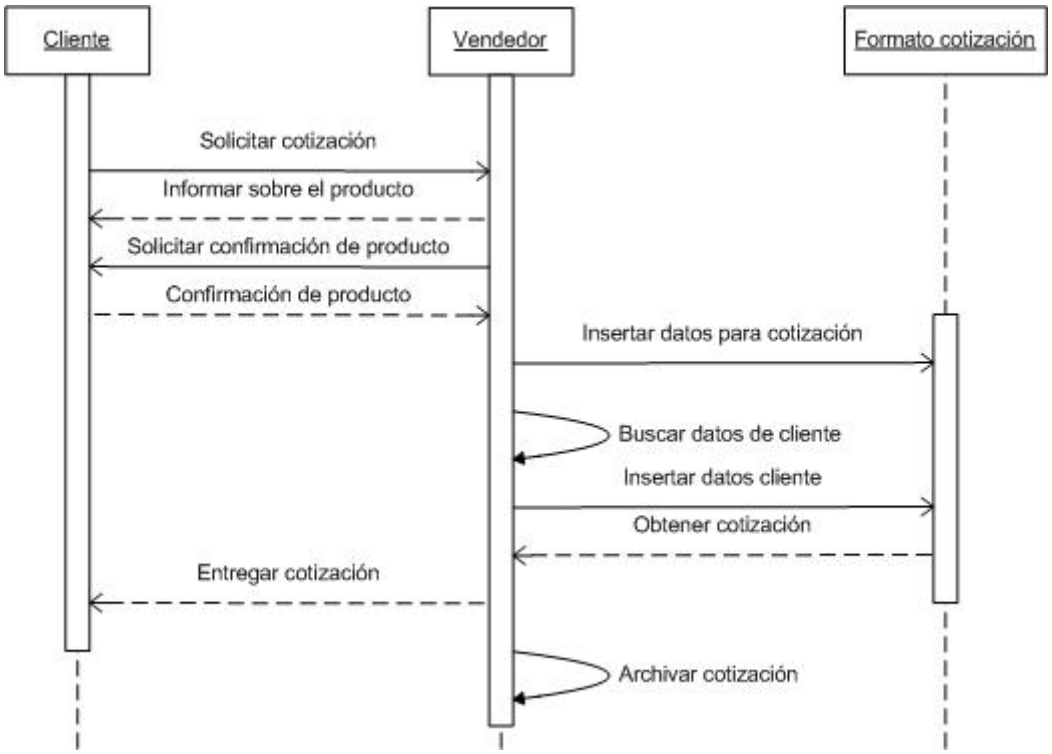


Figura VI.45. Diagrama de secuencia para el módulo de cotización

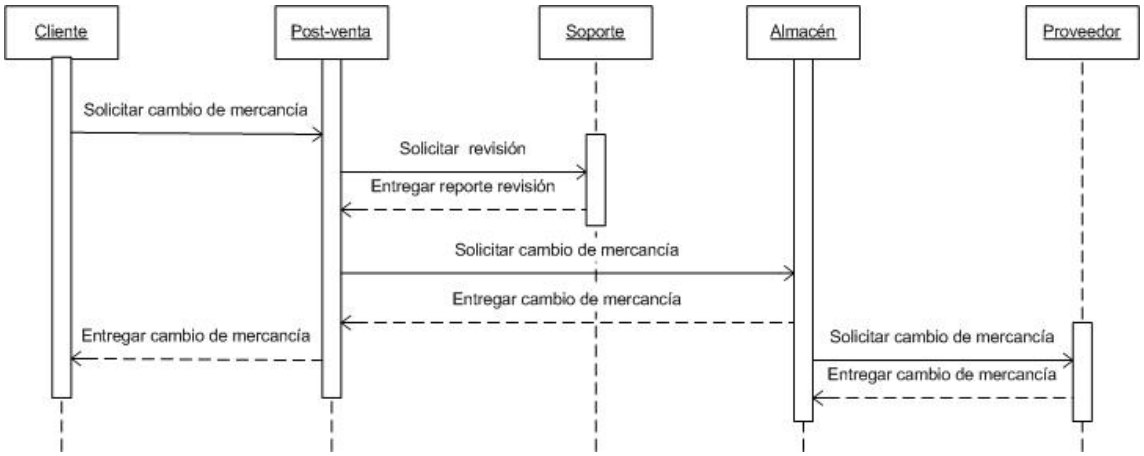


Figura VI.46. Diagrama de secuencia para el módulo de cambio de mercancía

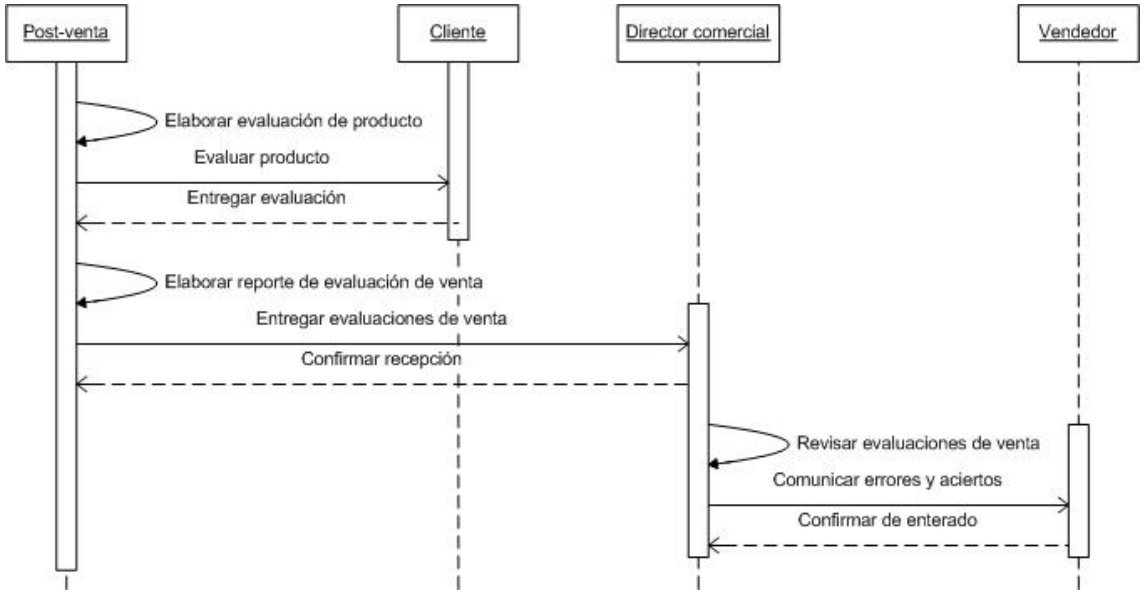


Figura VI.47. Diagrama de secuencia para el módulo de seguimiento de post-venta

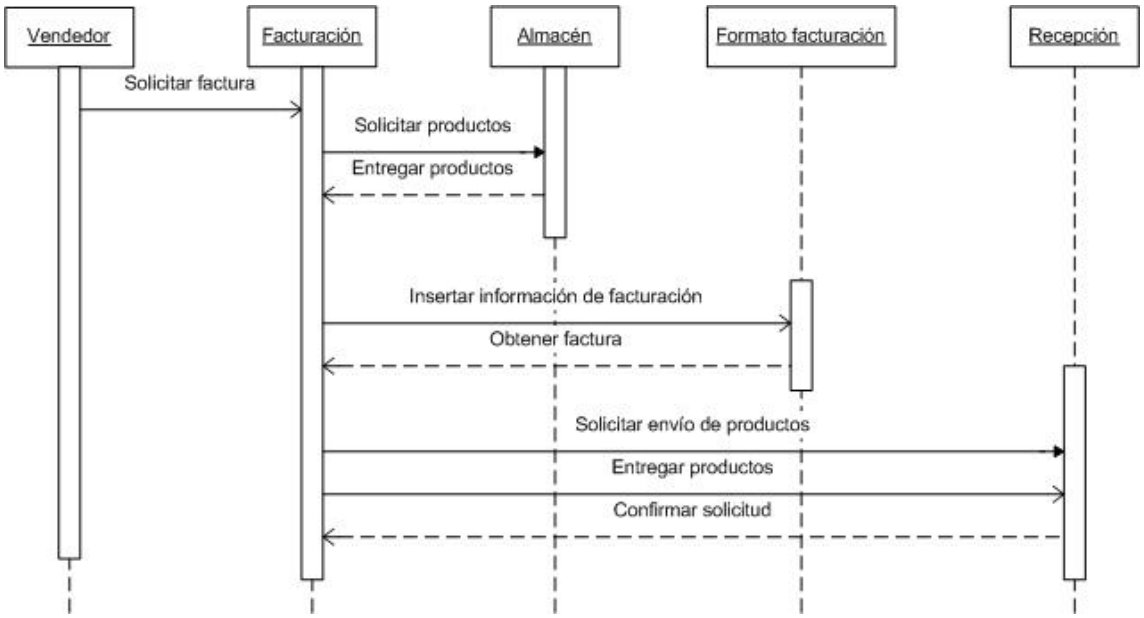


Figura VI.48. Diagrama de secuencia para el módulo de facturación

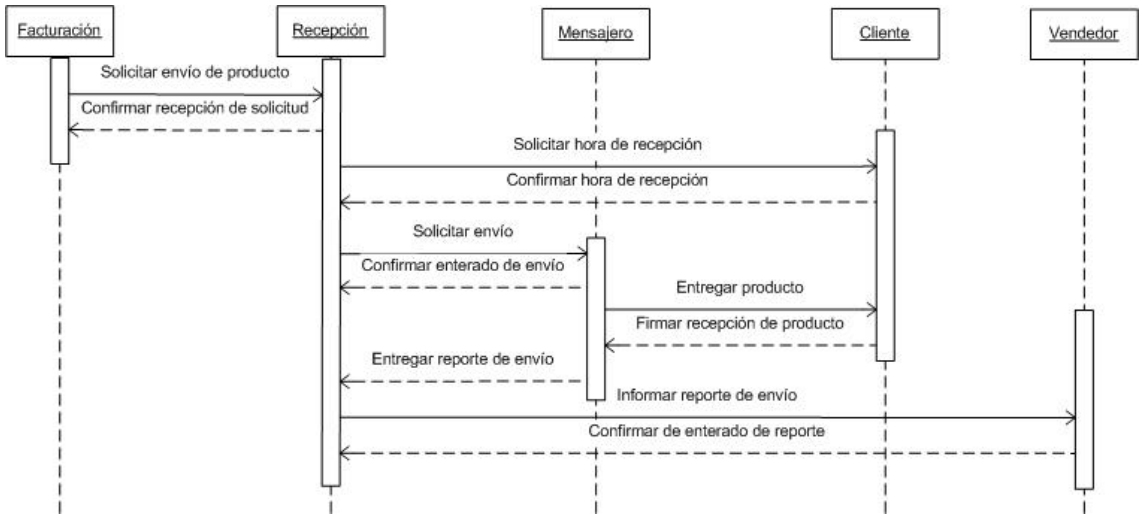


Figura VI.49. Diagrama de secuencia para el módulo de envío de productos

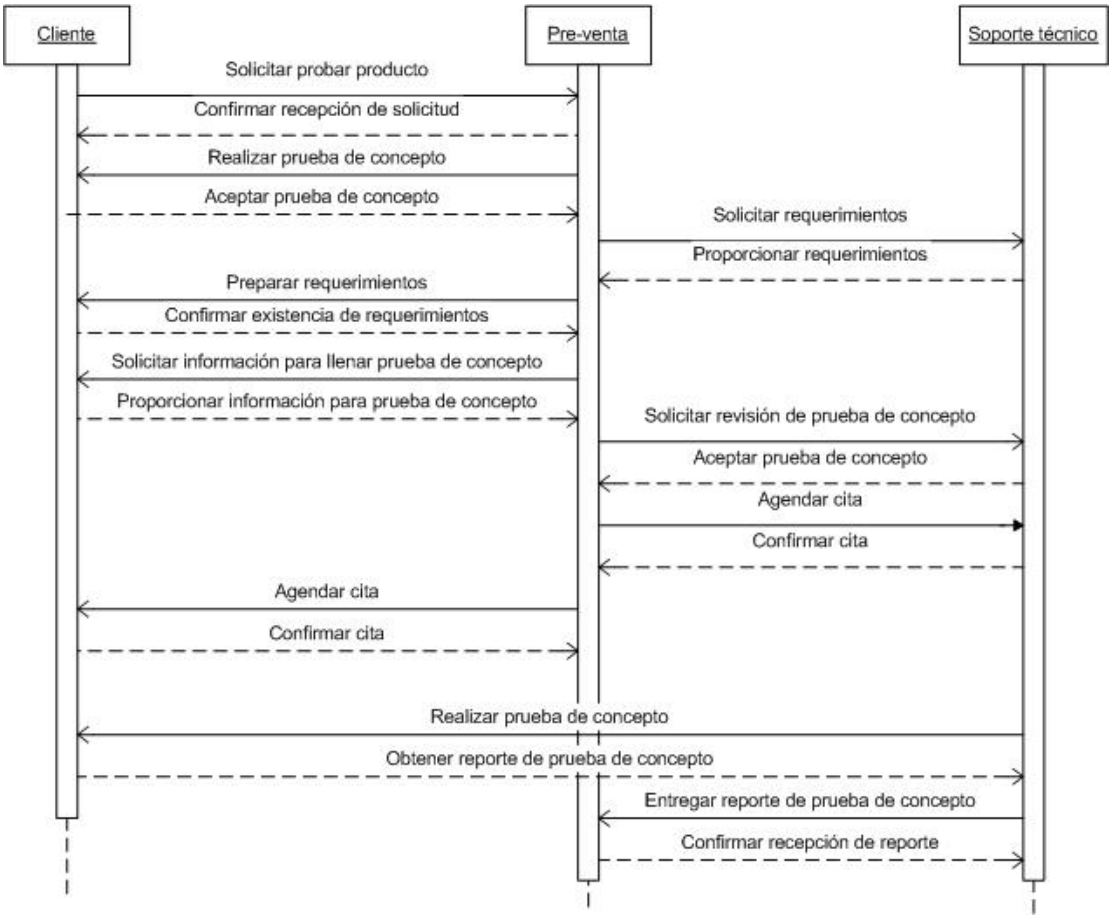


Figura VI.50. Diagrama de secuencia para el módulo de prueba de concepto

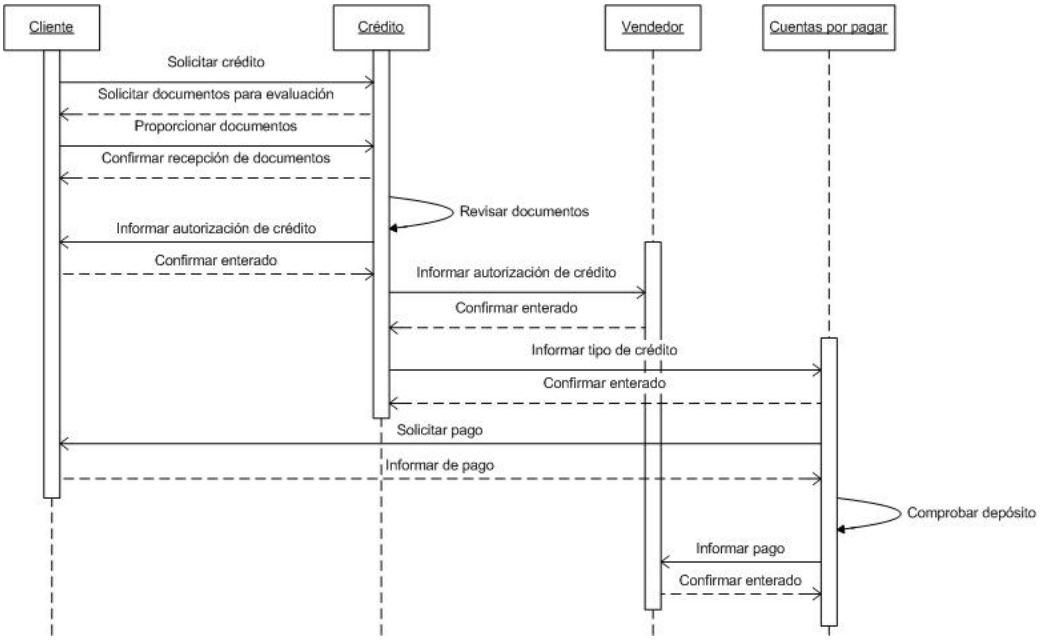


Figura VI.51. Diagrama de secuencia para el módulo de autorización de crédito

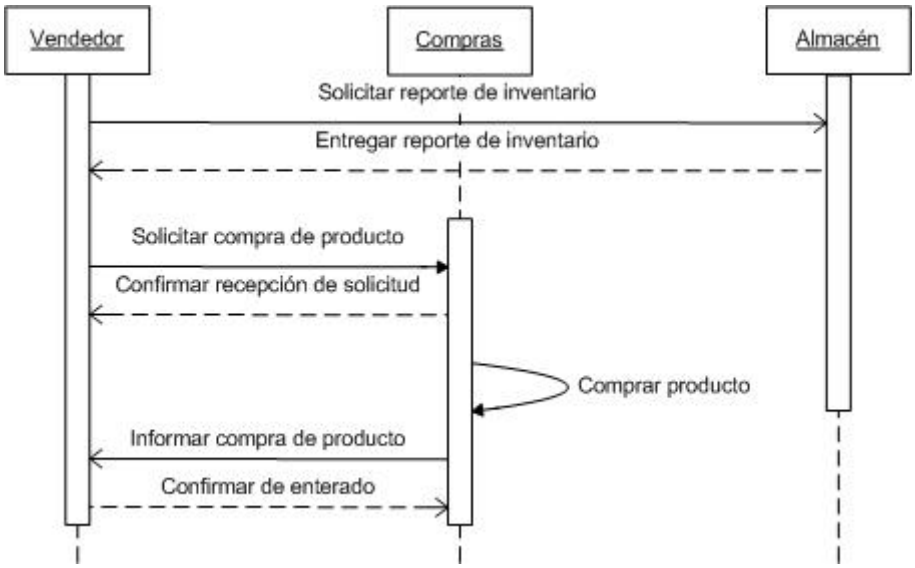


Figura VI.52. Diagrama de secuencia para el módulo de pedido interno de productos

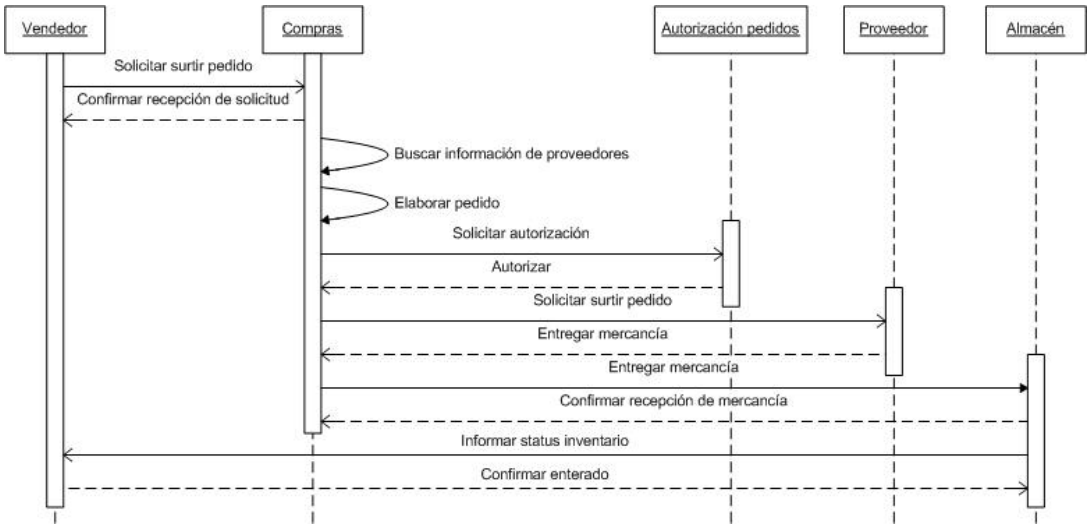


Figura VI.53. Diagrama de secuencia para el módulo de compras

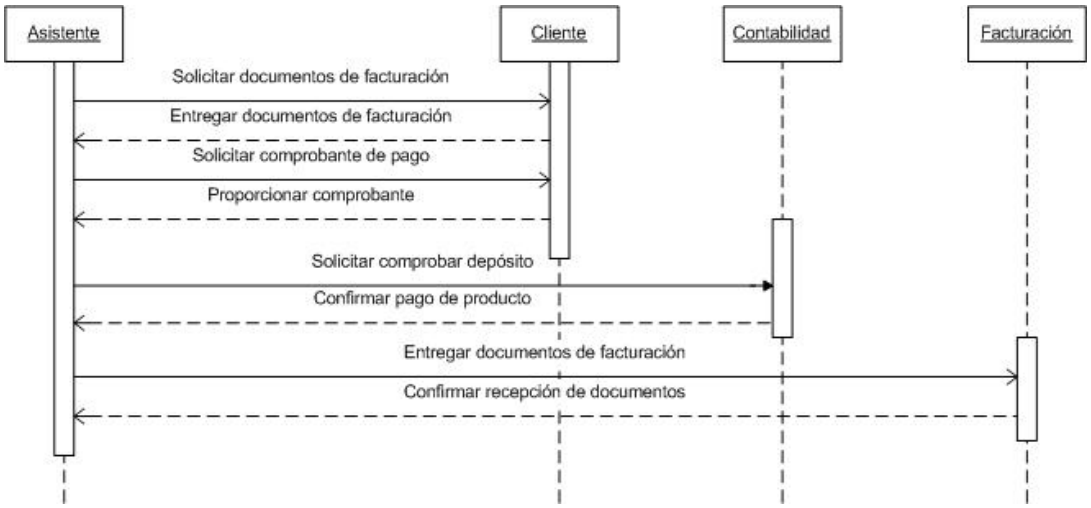


Figura VI.54. Diagrama de secuencia para el módulo de pre-facturación

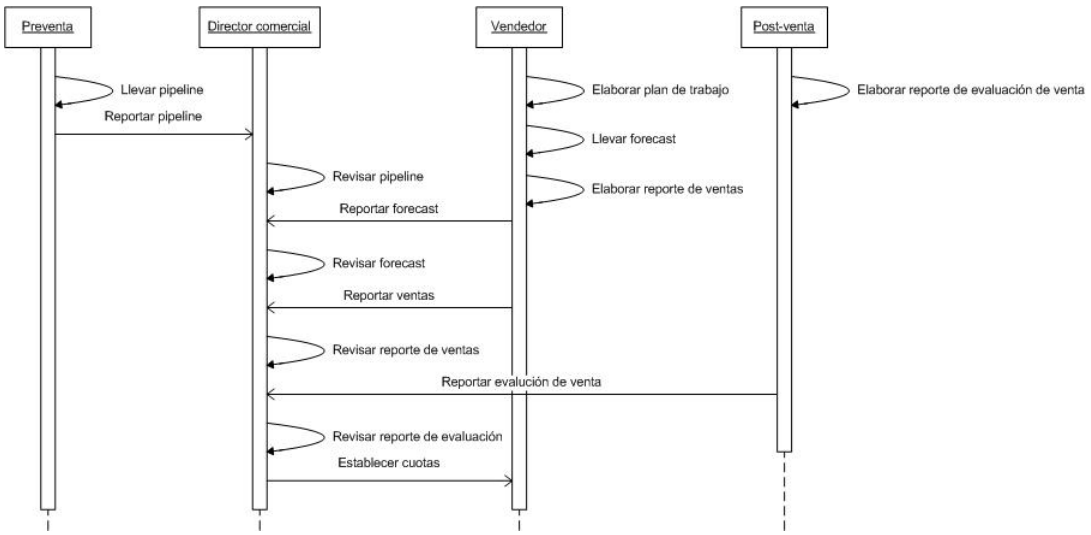


Figura VI.55. Diagrama de secuencia para el módulo de fuerza de ventas

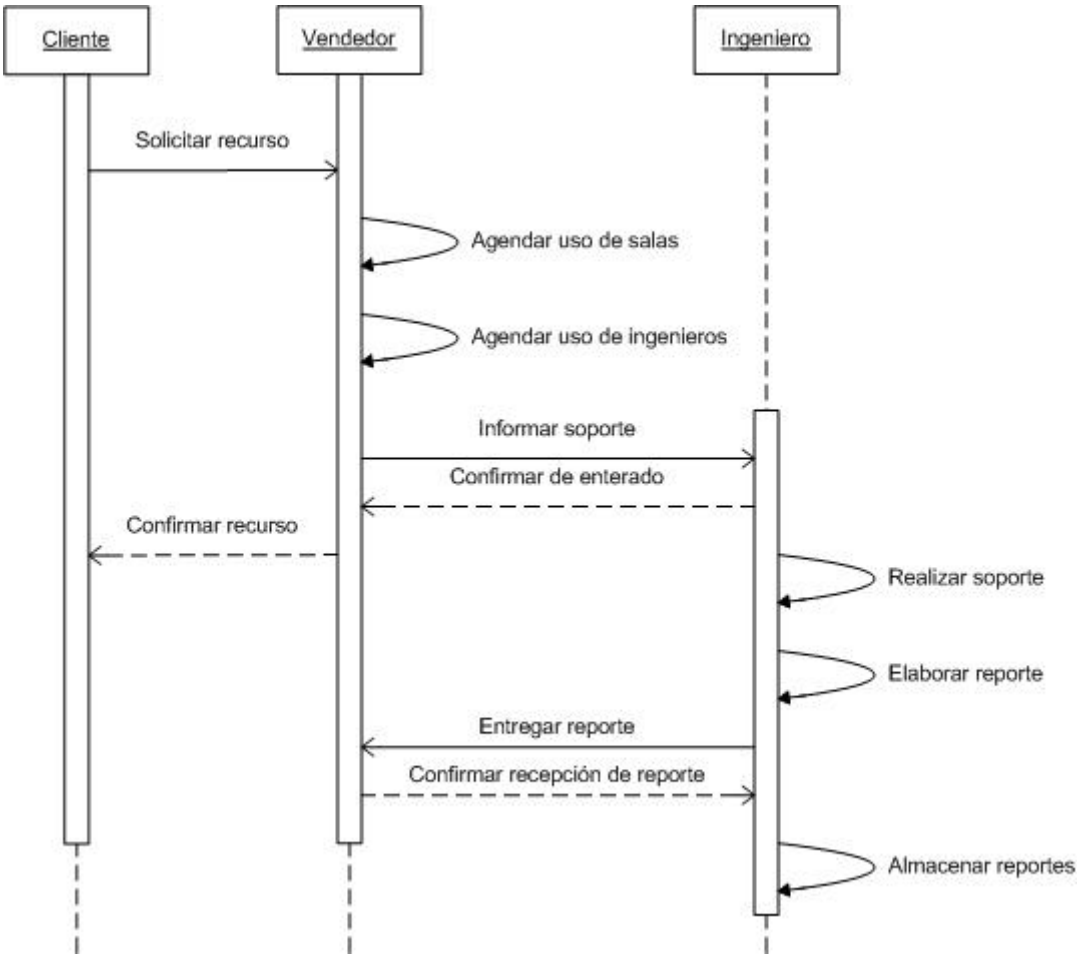


Figura VI.56. Diagrama de secuencia para el módulo de administración de recursos

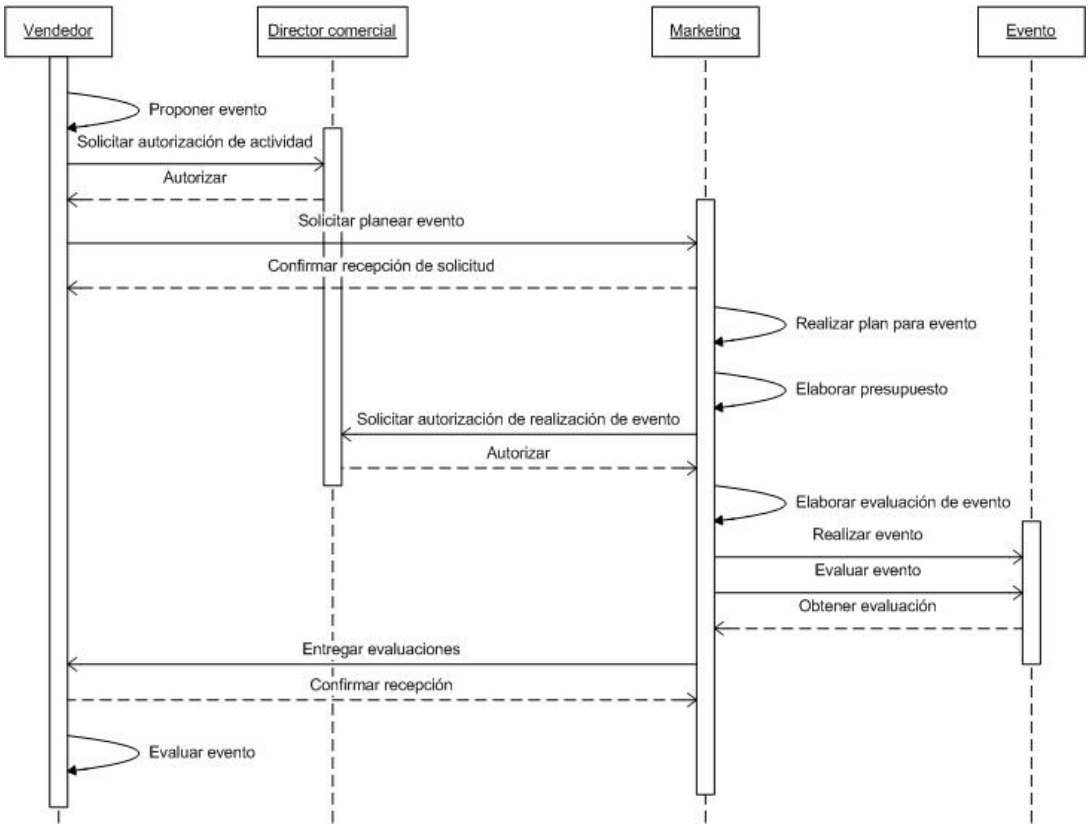


Figura VI.57. Diagrama de secuencia para el módulo de eventos

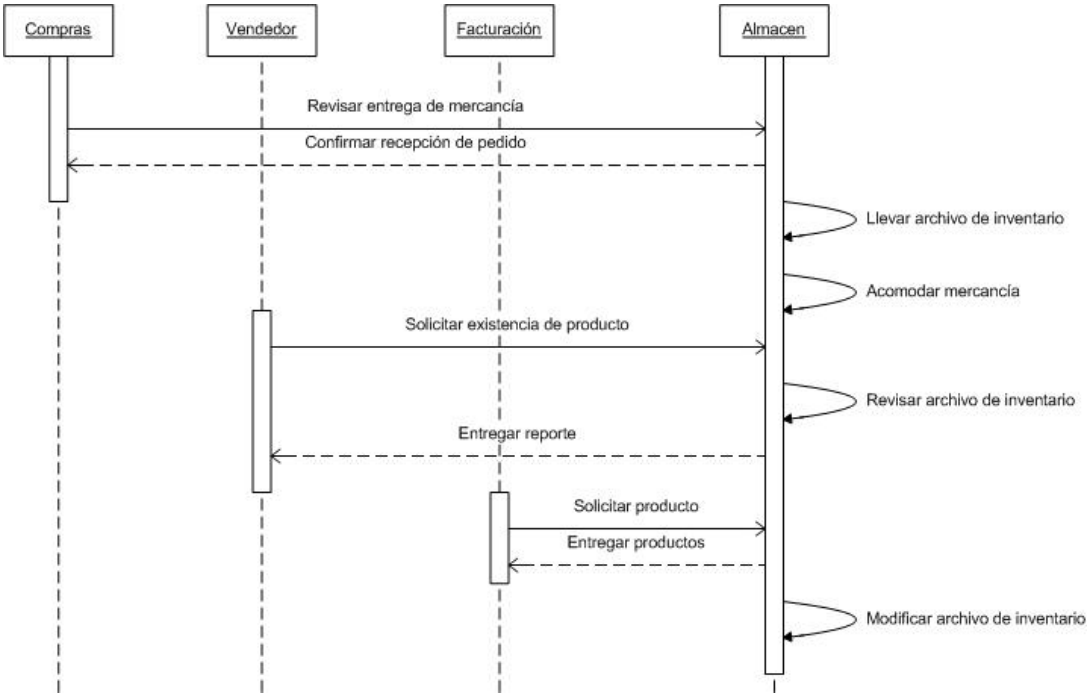


Figura VI.58. Diagrama de secuencia para el módulo de almacén

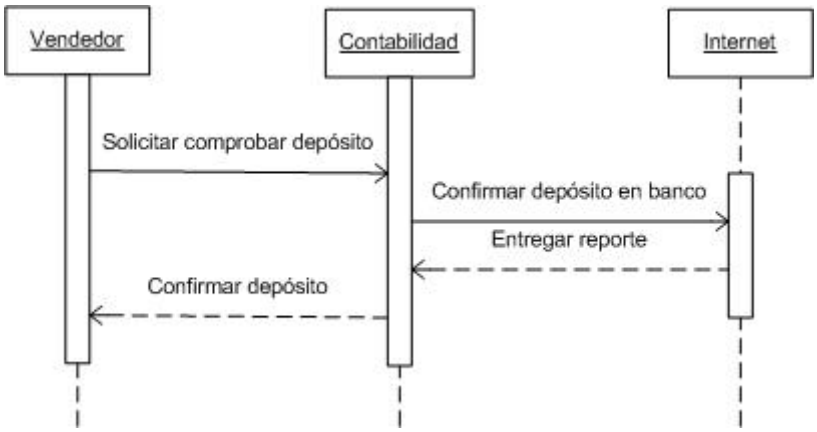


Figura VI.59. Diagrama de secuencia para el módulo de pago de mercancía

En base a los diagramas de secuencia mostrados anteriormente y después de realizar el análisis de éstos, se obtiene que hay procesos que pueden ser automatizados para agilizar y mejorar el proceso de ventas y con ello reducir los costos de operación de la empresa, por lo que los diagramas de secuencia que se muestran a continuación de la figura VI.60 a la figura VI.74, proporcionan la vista de estas automatizaciones.

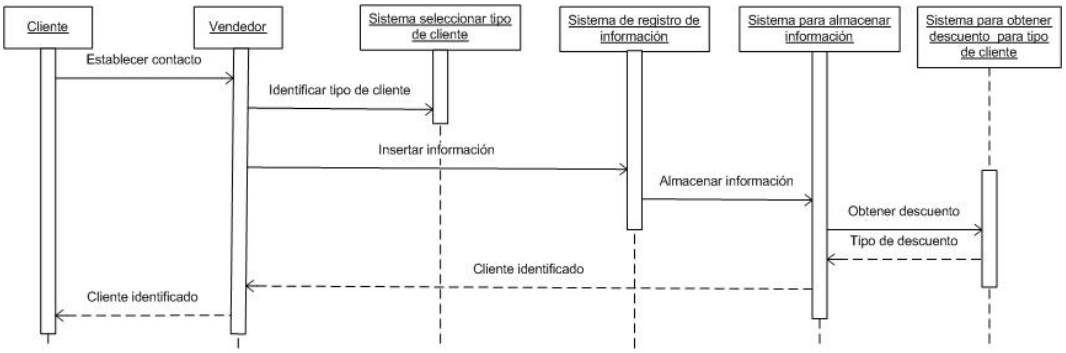


Figura VI.60. Diagrama de secuencia automatizado para el módulo de análisis de tipo de cliente

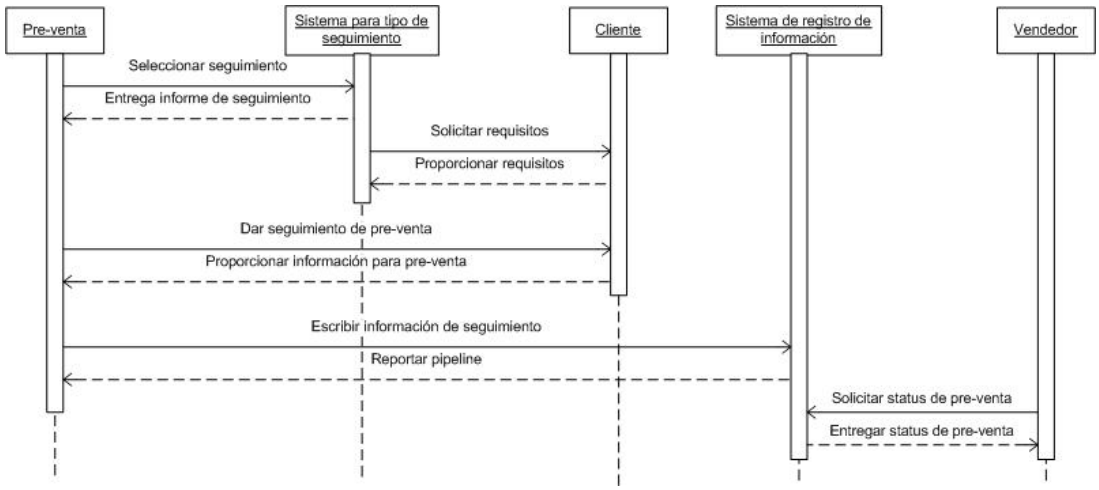


Figura VI.61. Diagrama de secuencia automatizado para el módulo de seguimiento de pre-venta

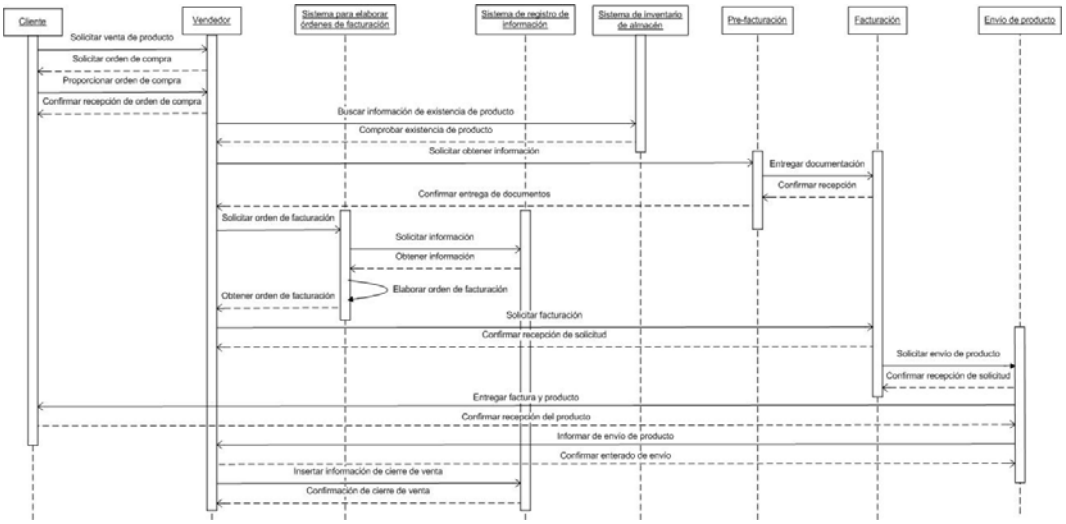


Figura VI.62. Diagrama de secuencia automatizado para el módulo de venta

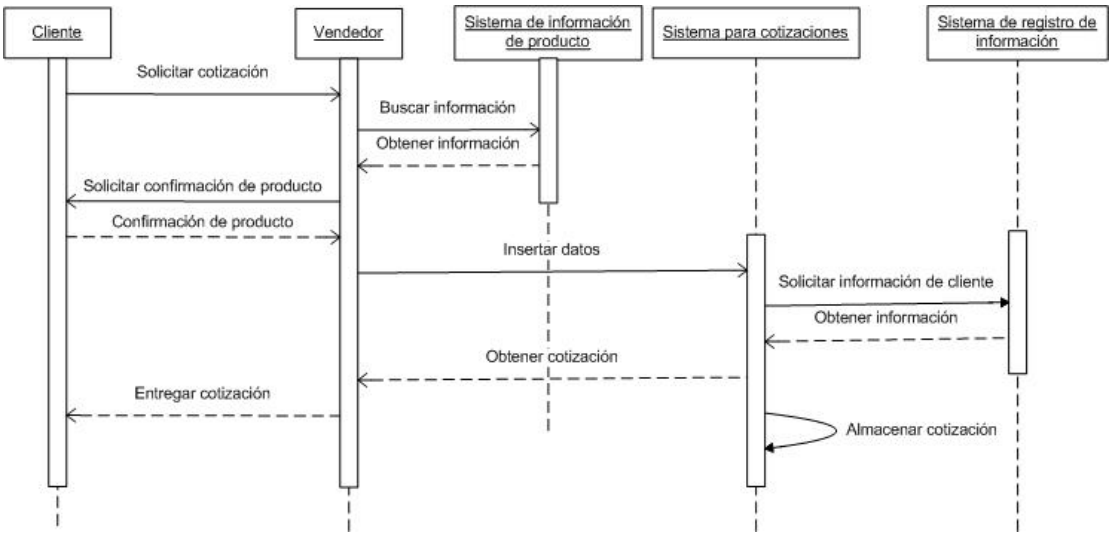


Figura VI.63. Diagrama de secuencia automatizado para el módulo de cotización

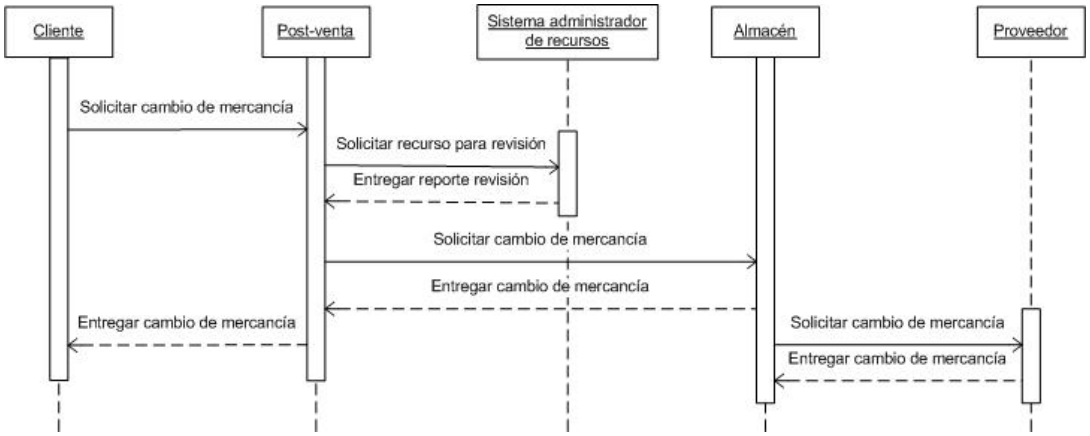


Figura VI.64. Diagrama de secuencia automatizado para el módulo de cambio de mercancía

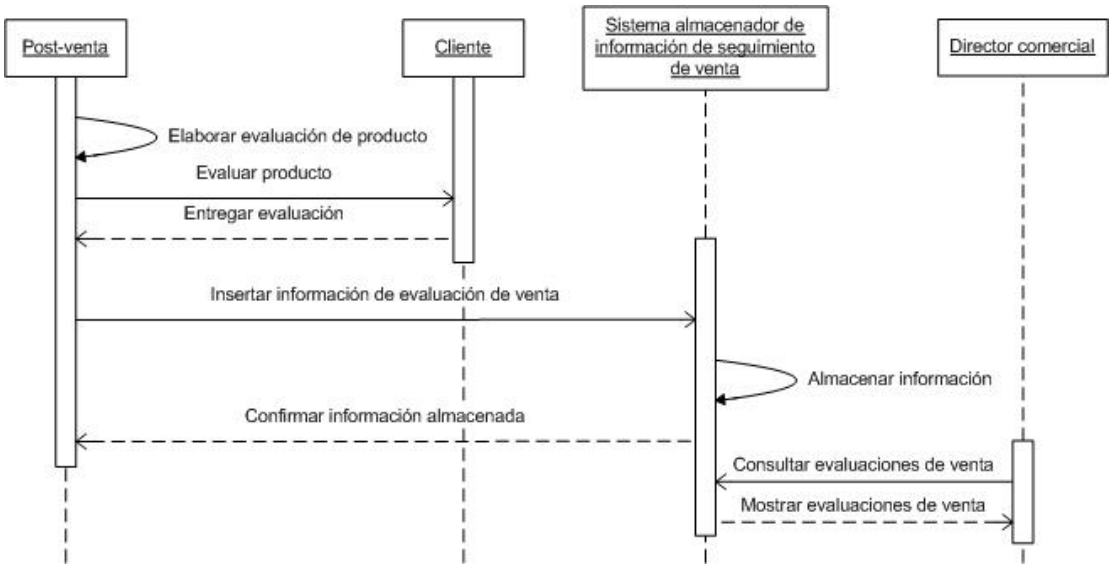


Figura VI.65. Diagrama de secuencia automatizado para el módulo de seguimiento de post-venta

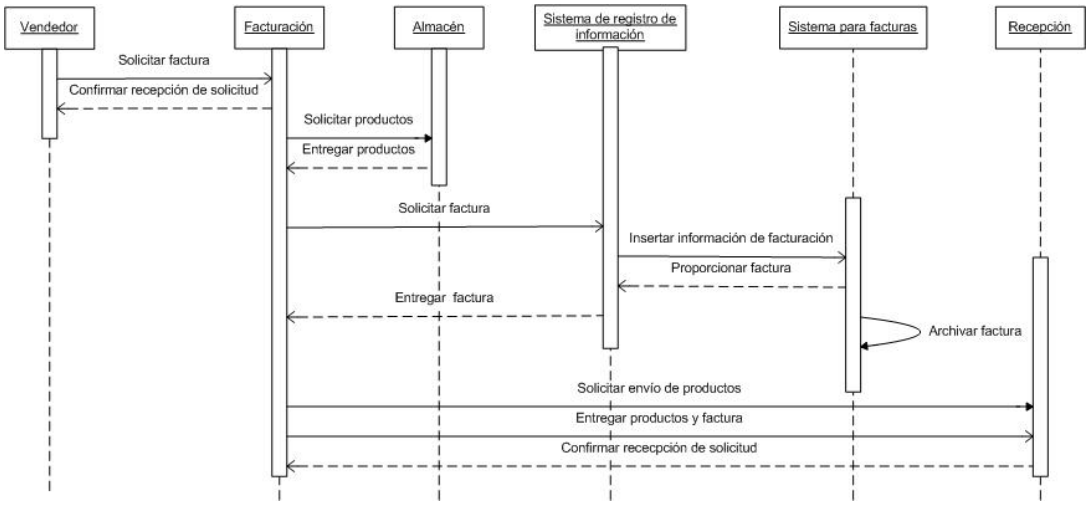


Figura VI.66. Diagrama de secuencia automatizado para el módulo de facturación

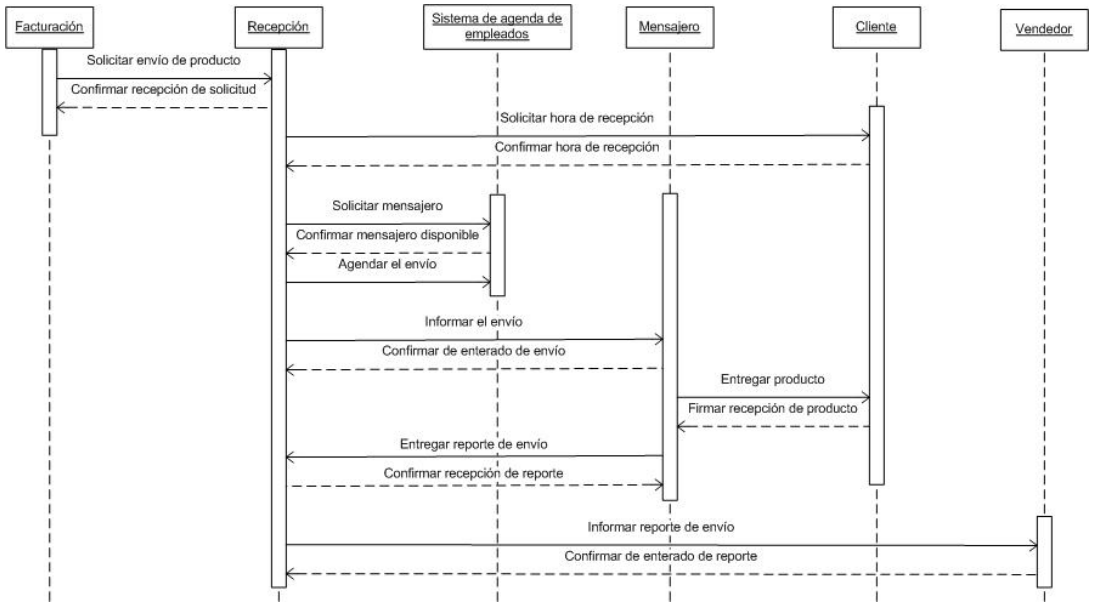


Figura VI.67. Diagrama de secuencia automatizado para el módulo de envío de productos

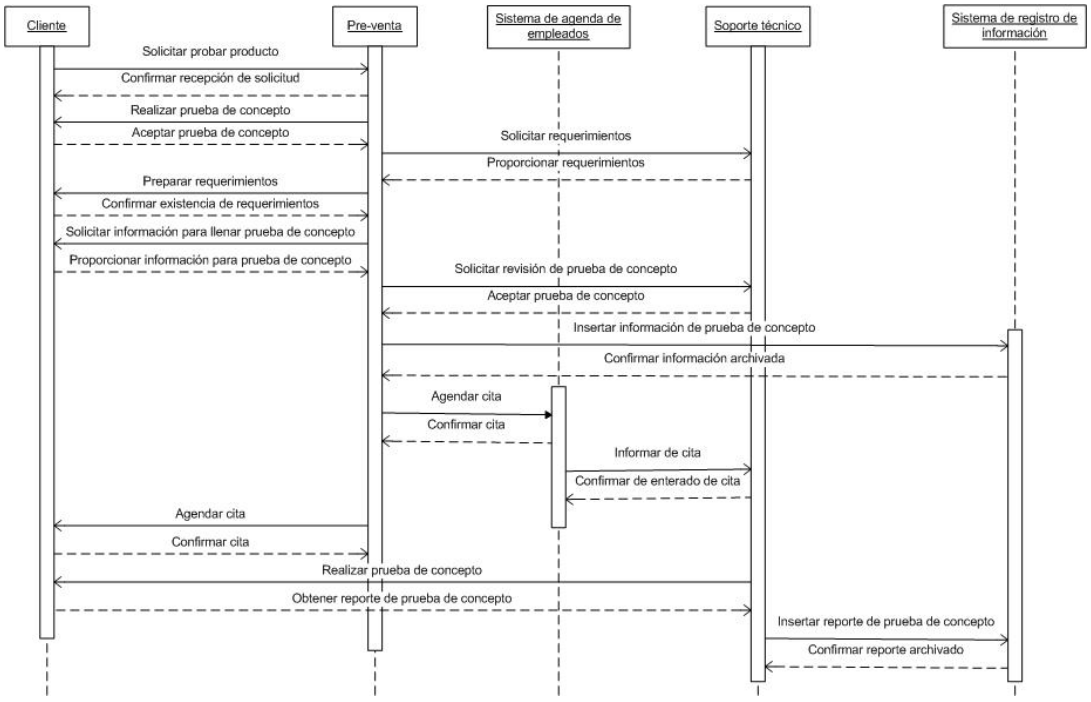


Figura VI.68. Diagrama de secuencia automatizado para el módulo de prueba de concepto

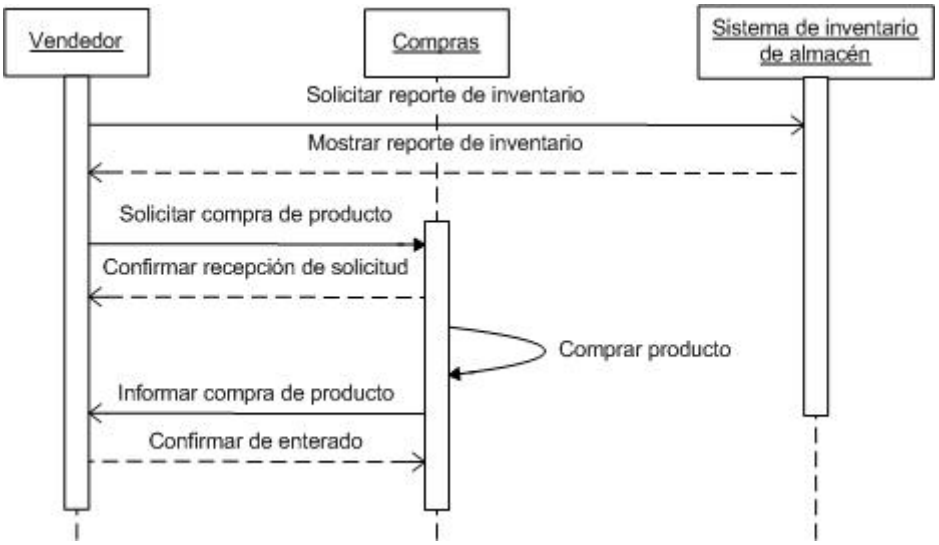


Figura VI.69. Diagrama de secuencia automatizado para el módulo de pedido interno de productos

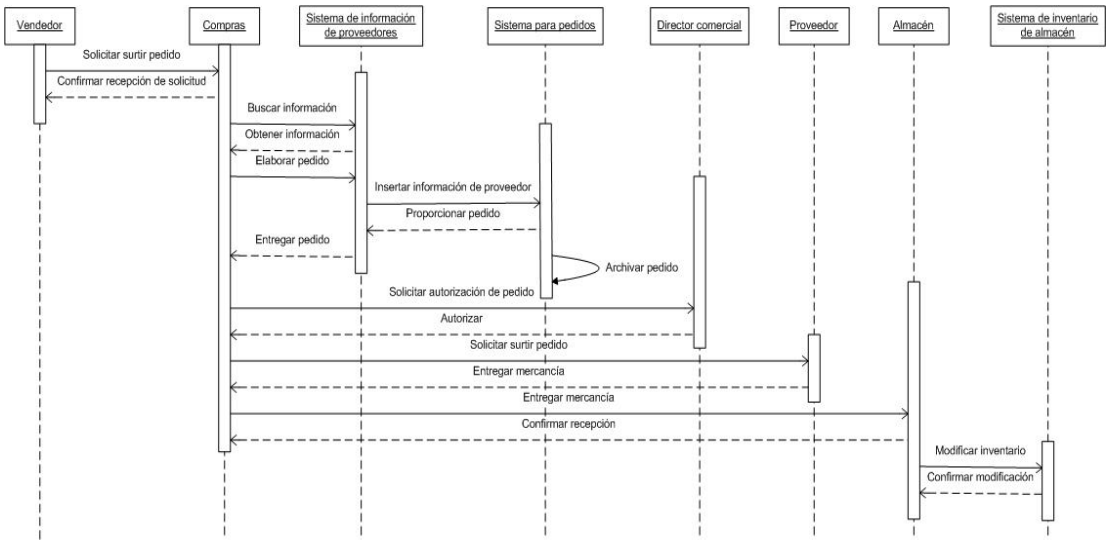


Figura VI.70. Diagrama de secuencia automatizado para el módulo de compras

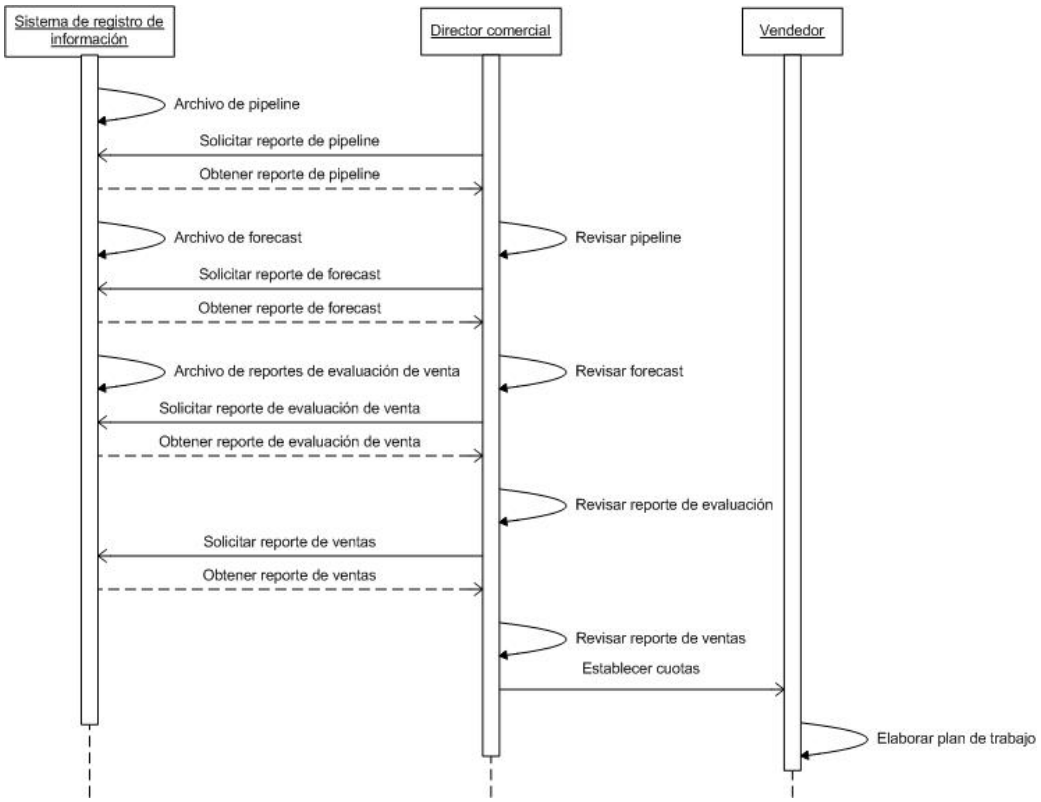


Figura VI.71. Diagrama de secuencia automatizado para el módulo de fuerza de ventas

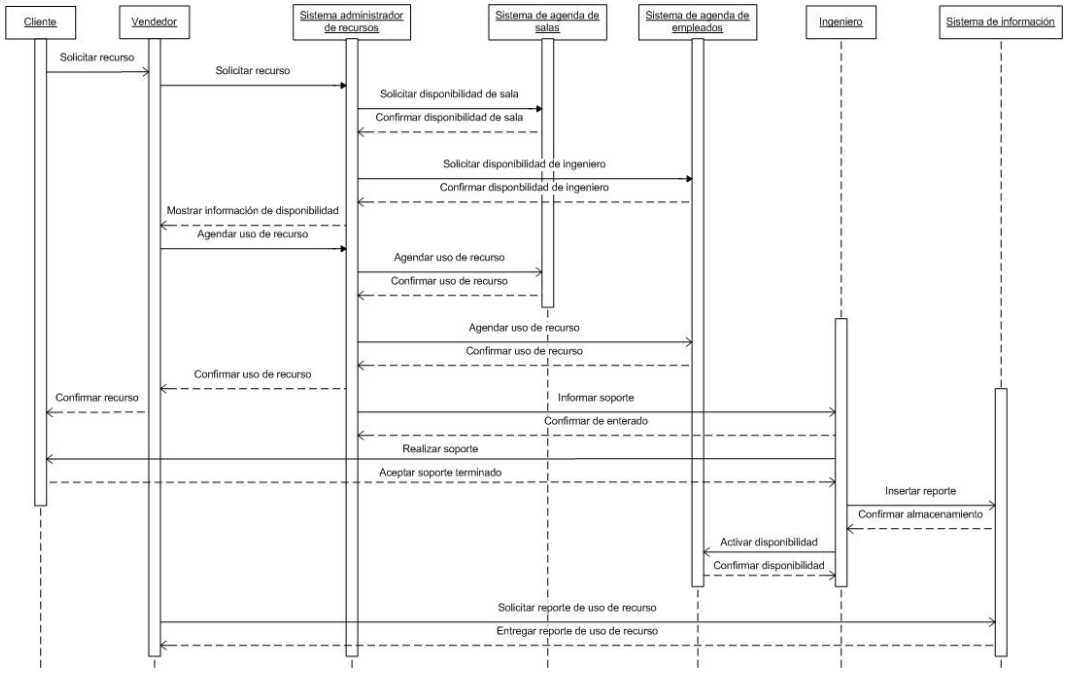


Figura VI.72. Diagrama de secuencia automatizado para el módulo de administración de recursos

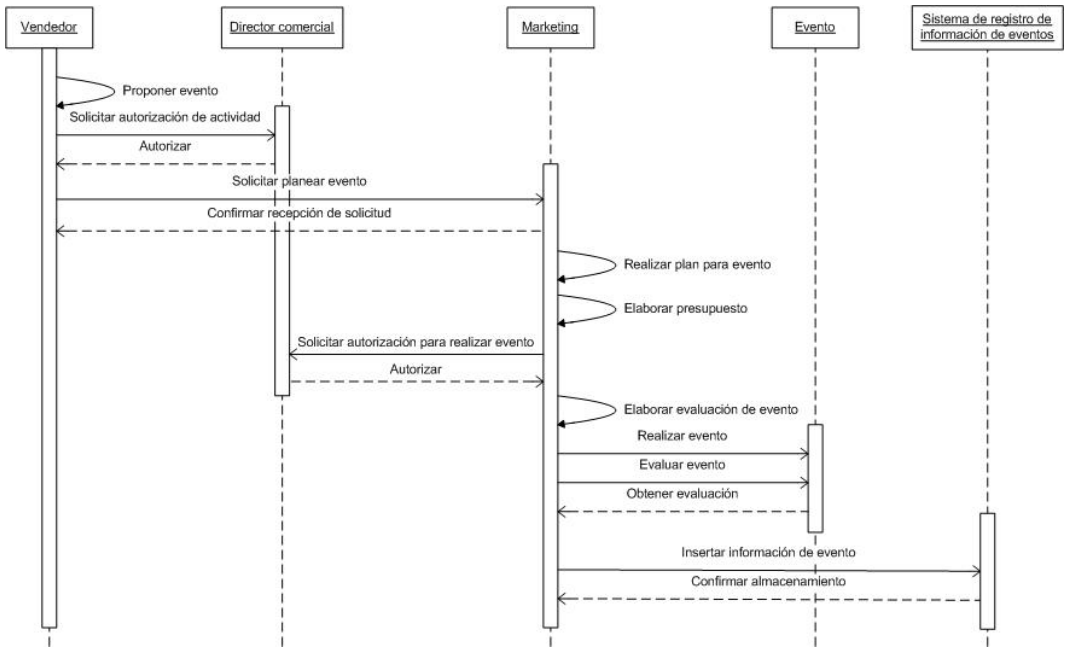


Figura VI.73. Diagrama de secuencia automatizado para el módulo de eventos

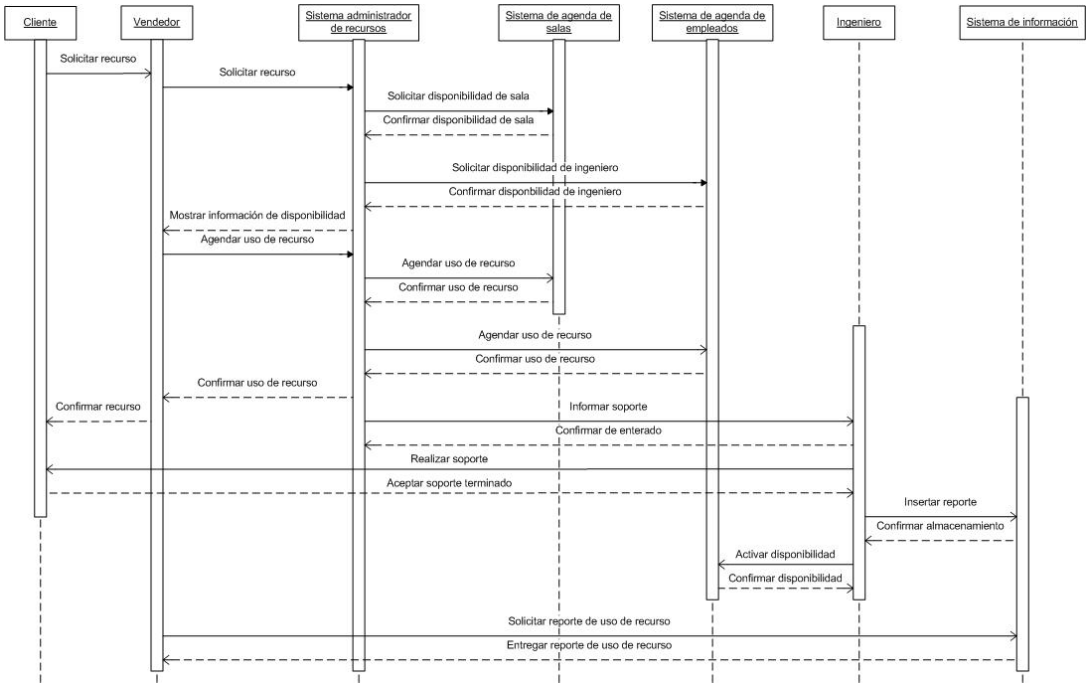


Figura VI.74. Diagrama de secuencia automatizado para el módulo de almacén

Analizando los diagramas de secuencia automatizados diseñados con el propósito de controlar los procesos del área de ventas, la empresa requiere implementar una solución que cubra el proceso completo tanto horizontal como vertical.

Hecho el análisis se tiene, que la solución debe abarcar cuando menos los siguientes módulos y procesos, que cubran los requerimientos de la empresa.

- Módulo de búsquedas
- Módulo de administración de información
- Módulo de administración de reportes
- Módulo de administración del proceso de ventas
- Módulo de administración de clientes
- Módulo de administración de formatos

Con el fin de organizar de manera general el modelo, a continuación se muestran los paquetes que corresponden a los subsistemas del sistema total de ventas, para que con ello se pueda obtener la interfaz correspondiente al sistema de ventas.

La figura VI.75, muestra el diagrama general de paquetes para el sistema de ventas propuesto.

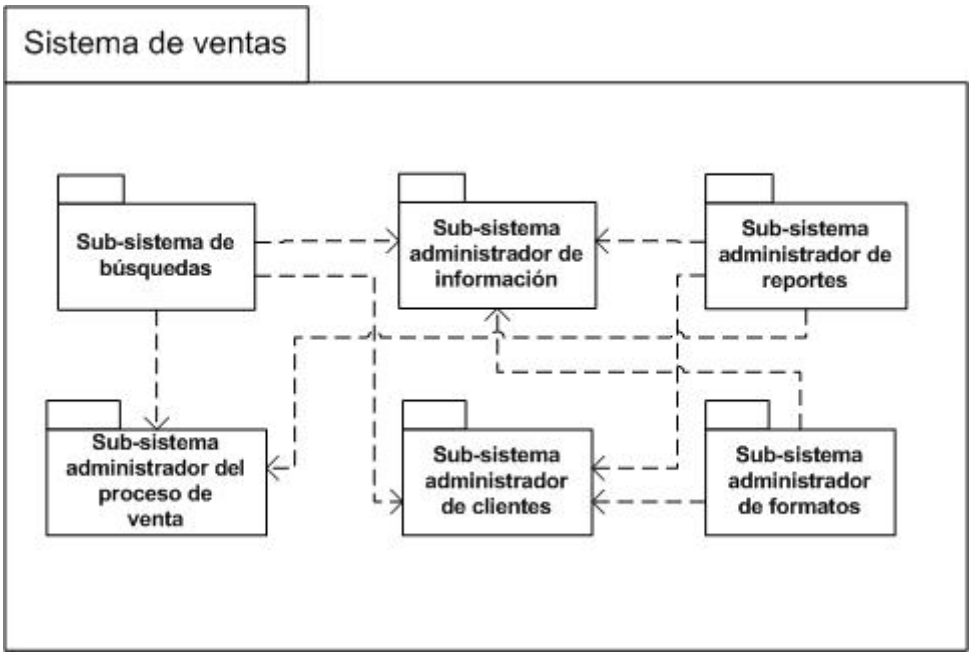


Figura VI.75. Diagrama general de paquetes para el sistema de ventas

En base a los subsistemas que deben formar el sistema de ventas mostrados en la figura VI.75, se tiene que de manera general, la interfaz que el usuario necesita para poder utilizar el sistema y de esta manera agilizar su proceso de ventas, es el que se muestra a continuación por medio de las siguientes pantallas que abarcan de la figura VI.76 a la figura VI.102.

Dichas pantallas son propuestas basándose en los subsistemas mencionados anteriormente, los cuales cubren con las propuestas de los diagramas de secuencia automatizados de utilizar un sistema en el proceso de ventas.

Se hace el análisis de forma tal que cada uno de los módulos propuestos contenga la información necesaria para la empresa, información que es recabada con el análisis del escenario, los objetos que lo forman, los casos de uso y los diagramas de secuencia.

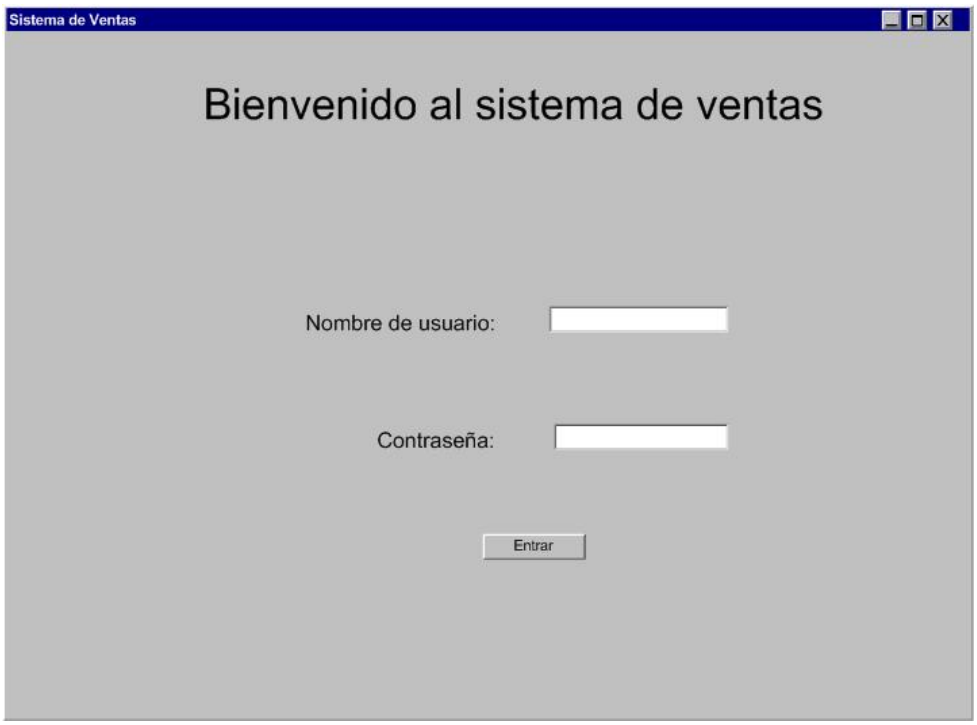


Figura VI.76. Pantalla de bienvenida del sistema de ventas

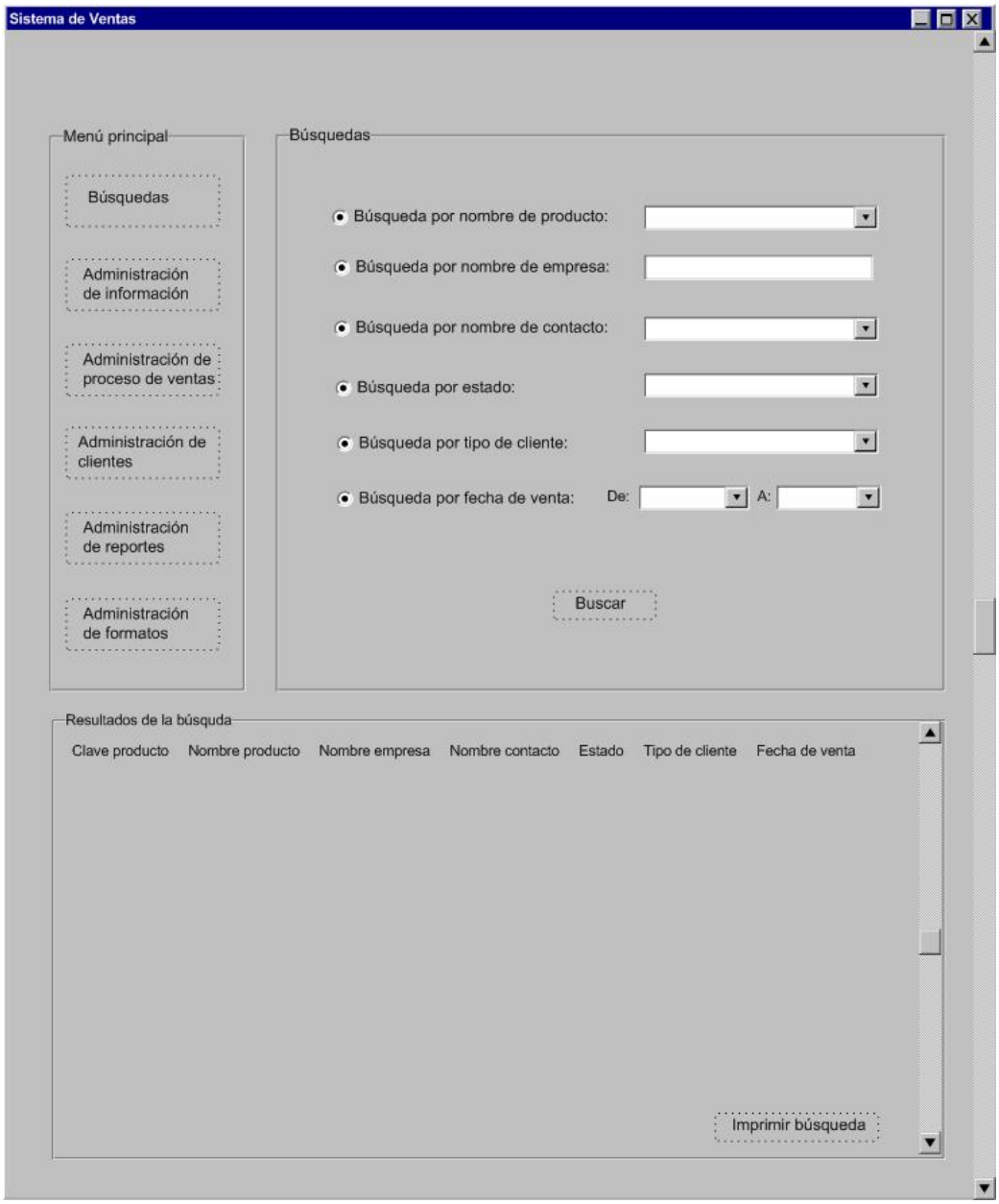


Figura VI.77. Pantalla de búsquedas del sistema de ventas

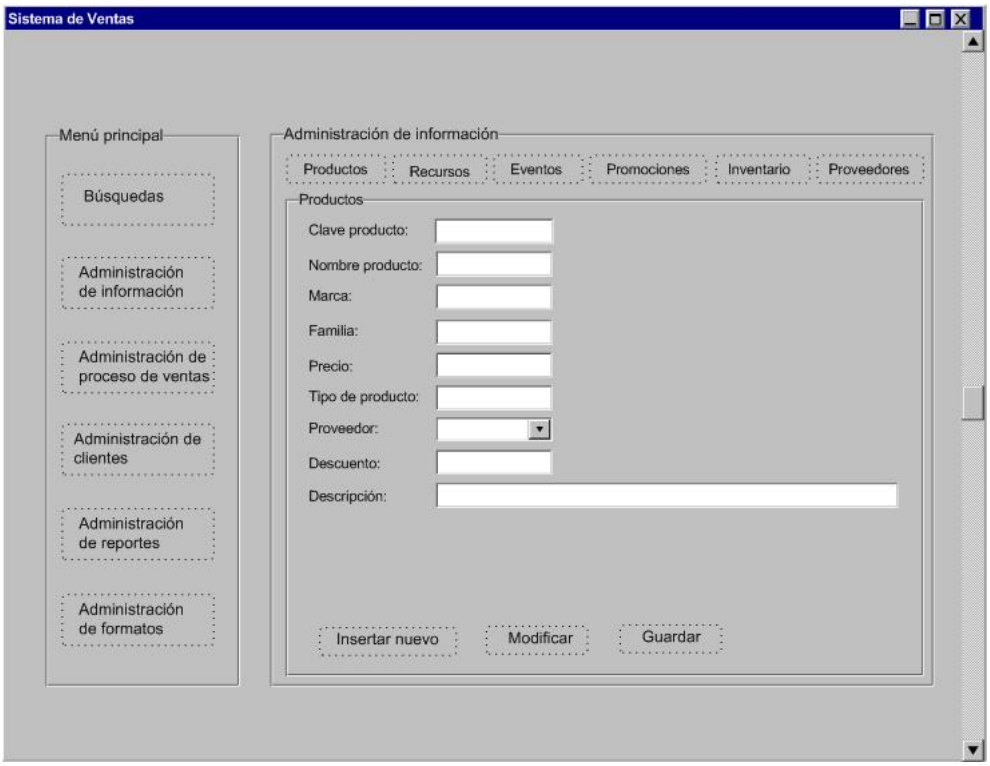


Figura VI.78. Pantalla de administración de información de productos

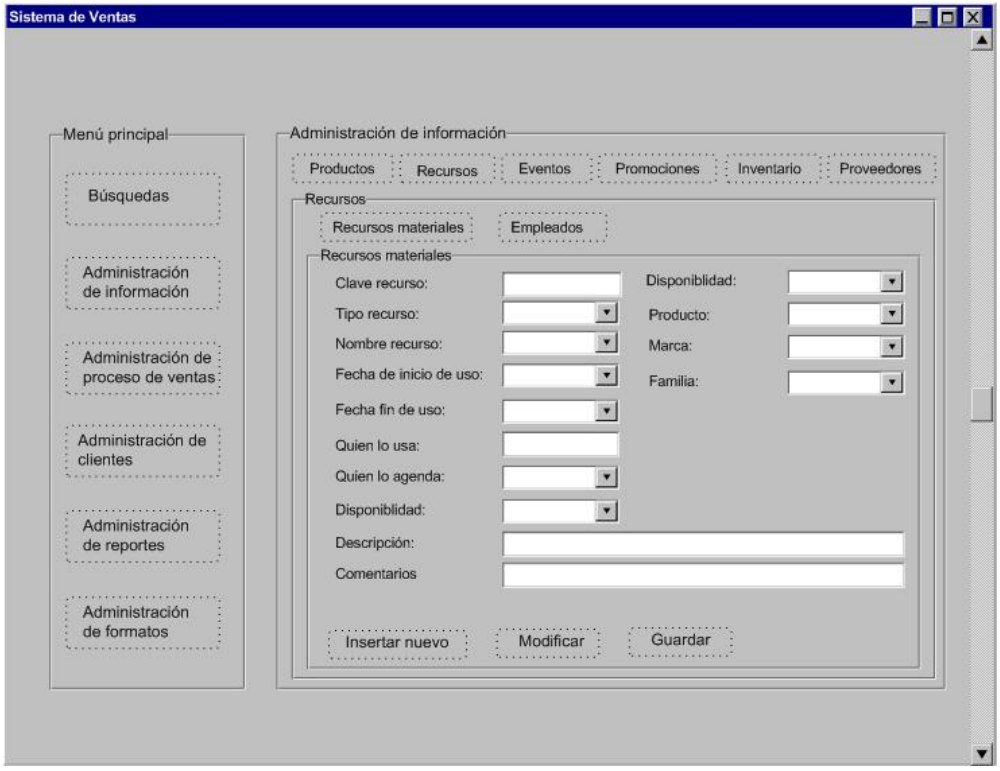


Figura VI.79. Pantalla de administración de información de recursos materiales

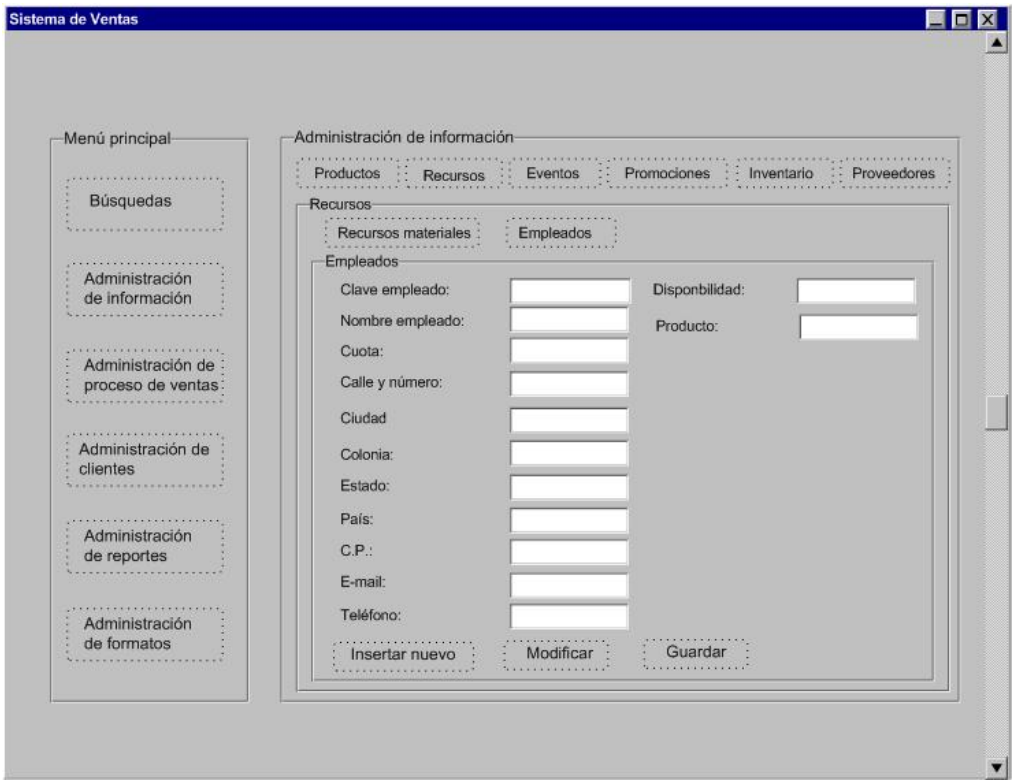


Figura VI.80. Pantalla de administración de información de recursos-empleados

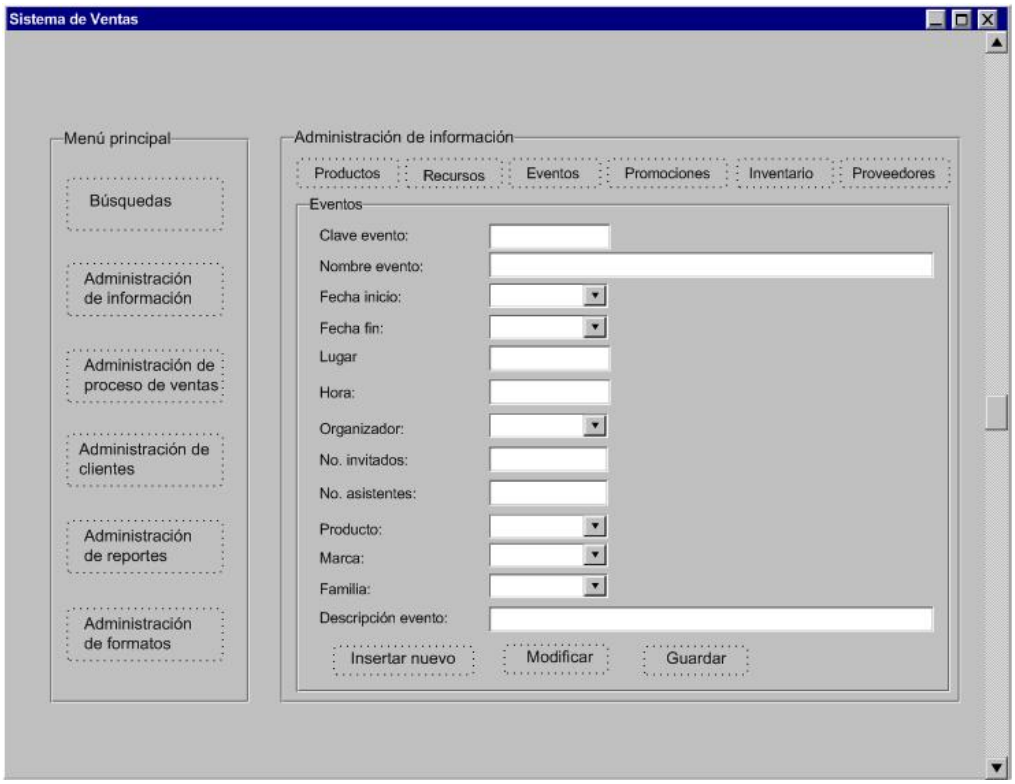


Figura VI.81. Pantalla de administración de información de eventos

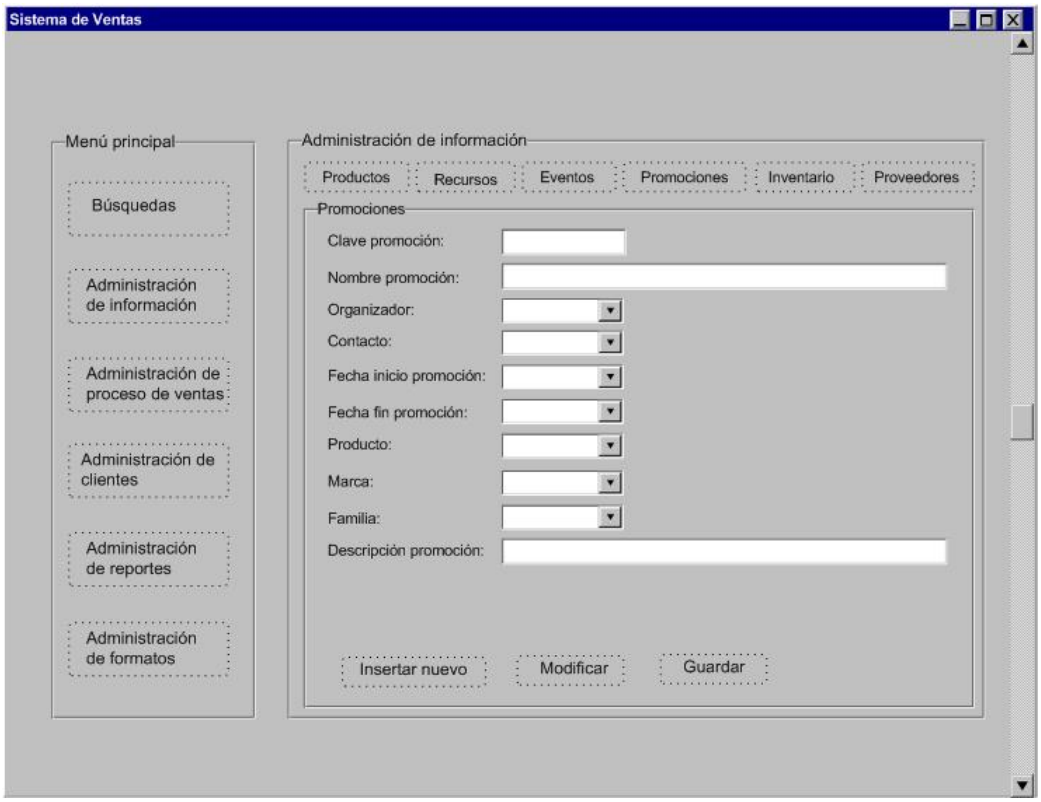


Figura VI.82. Pantalla de administración de información de promociones

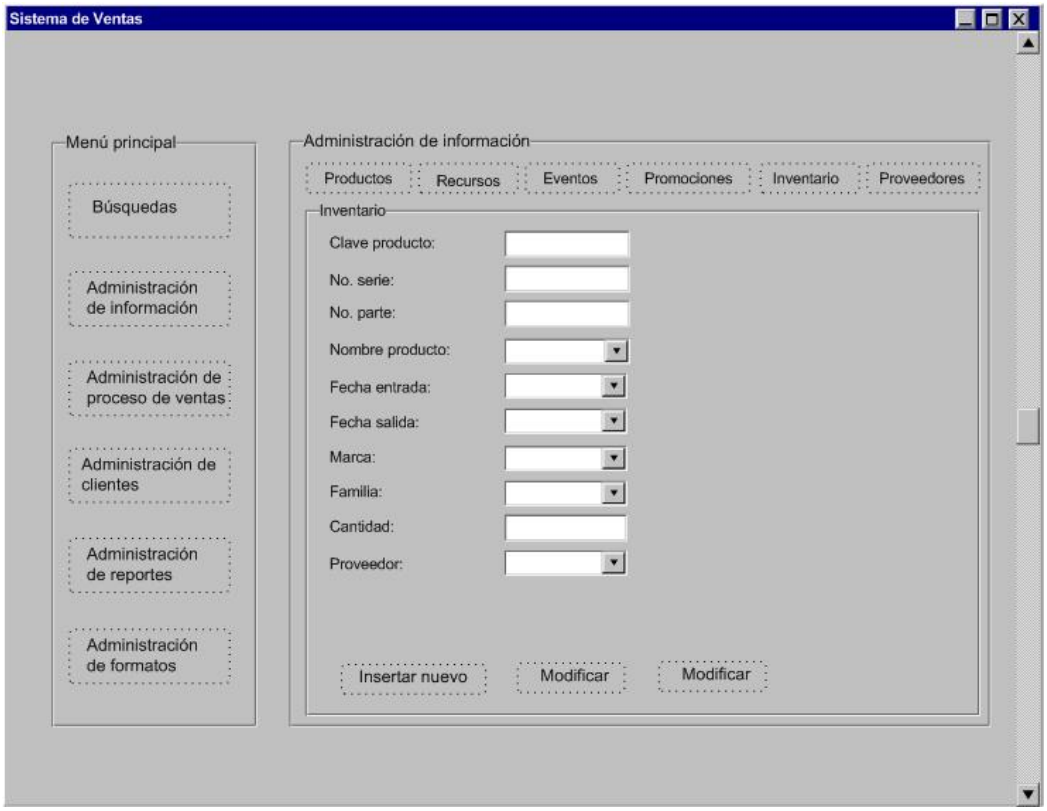


Figura VI.83. Pantalla de administración de información de inventario

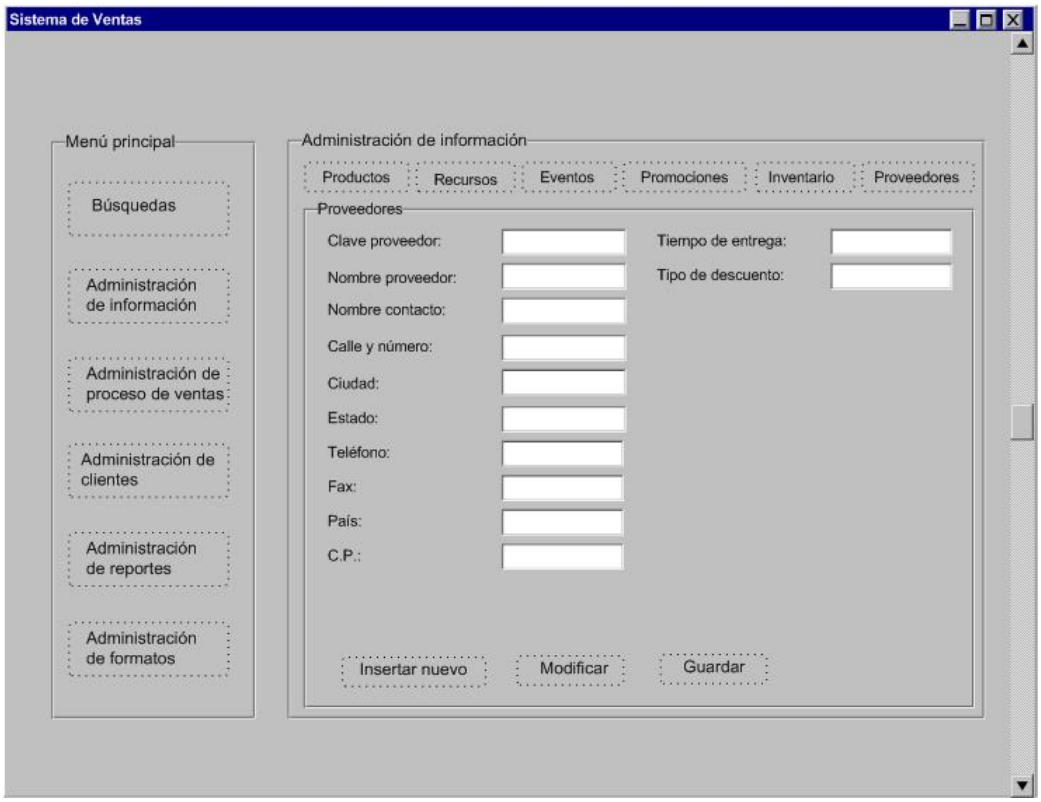


Figura VI.84. Pantalla de administración de información de proveedores

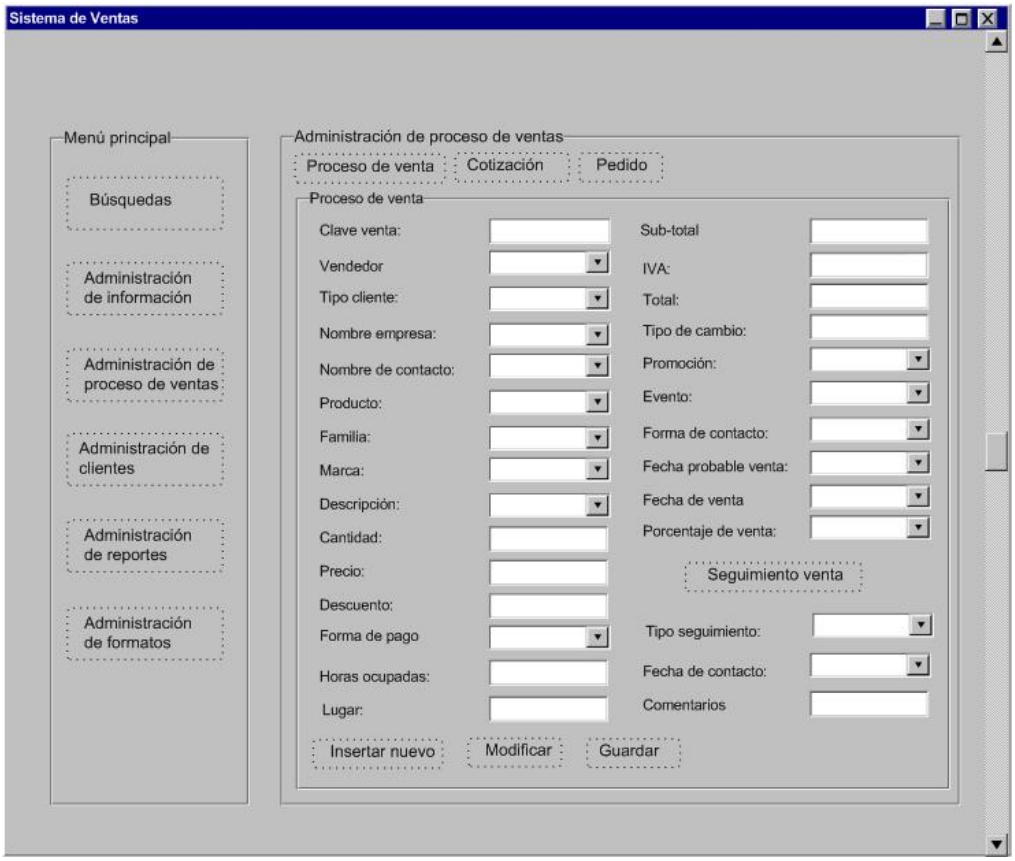


Figura VI.85. Pantalla de administración de proceso de ventas

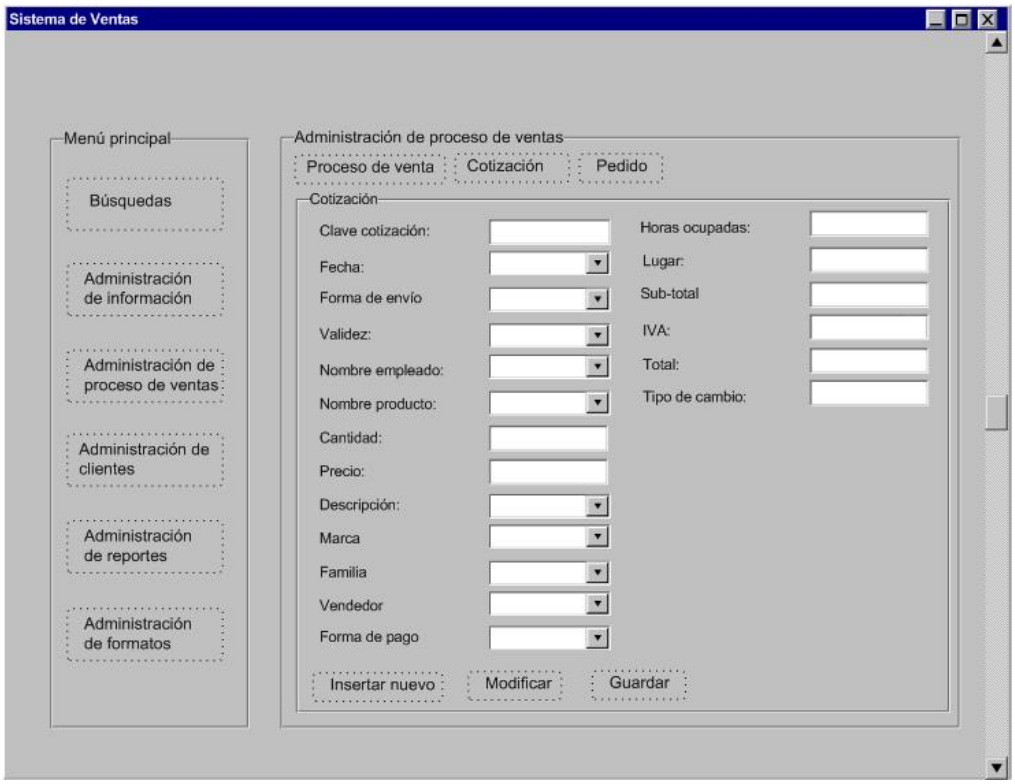


Figura VI.86. Pantalla de administración de proceso de ventas para cotizaciones

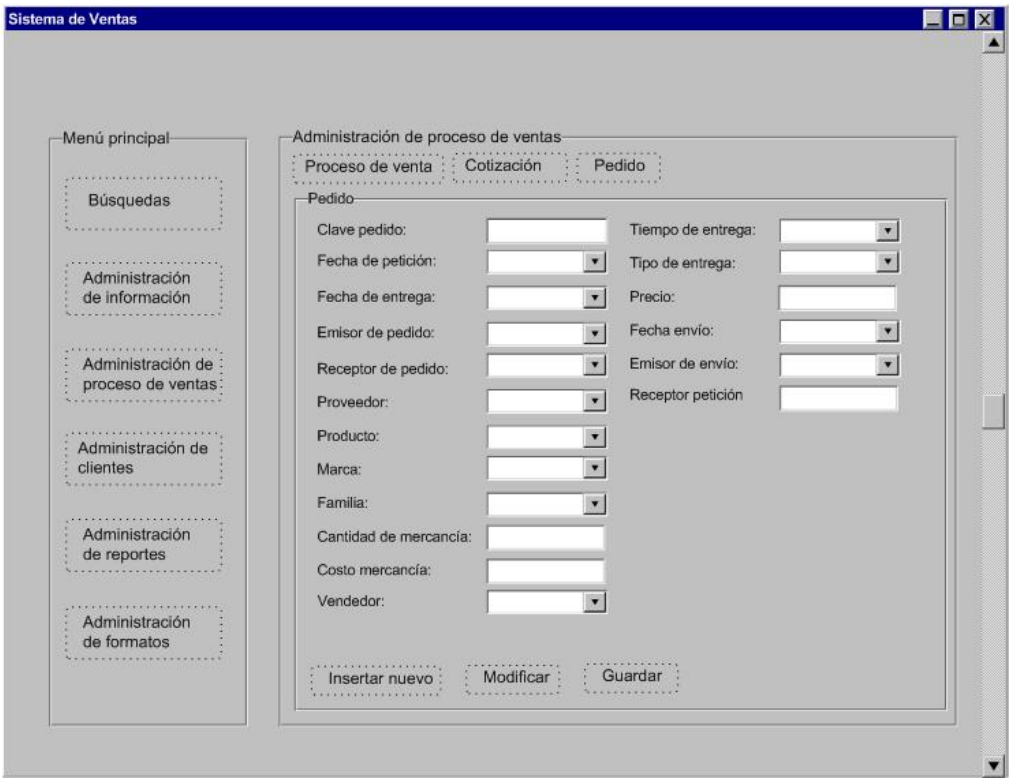


Figura VI.87. Pantalla de administración de proceso de ventas para pedidos

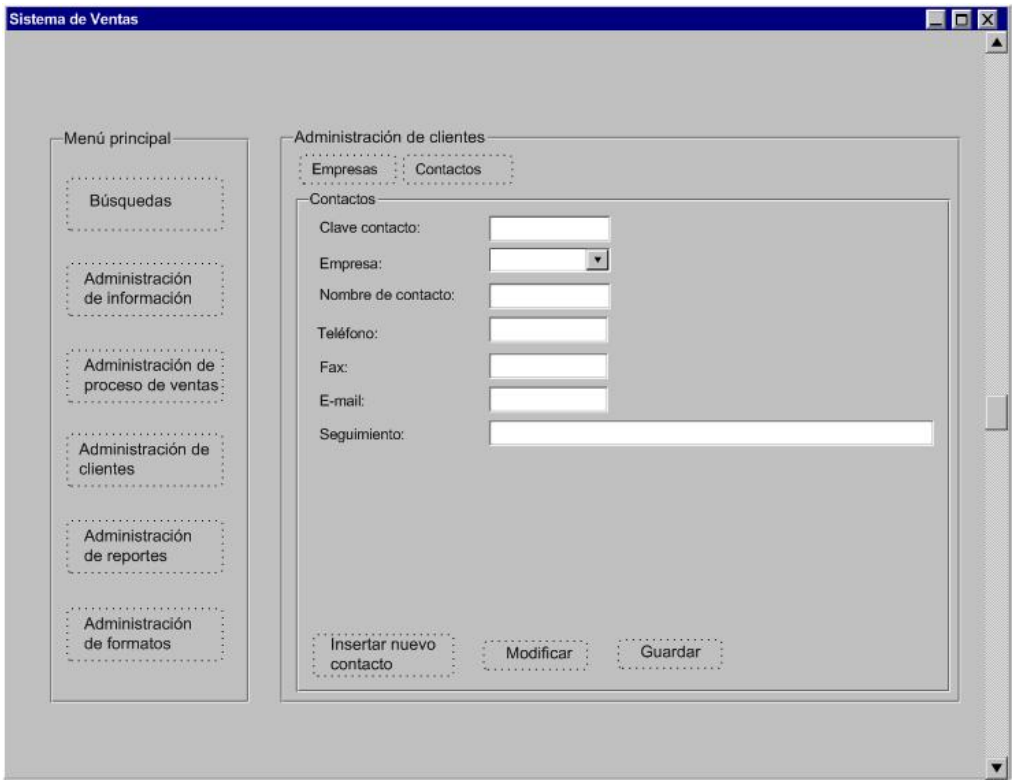


Figura VI.88. Pantalla de administración de clientes por empresa

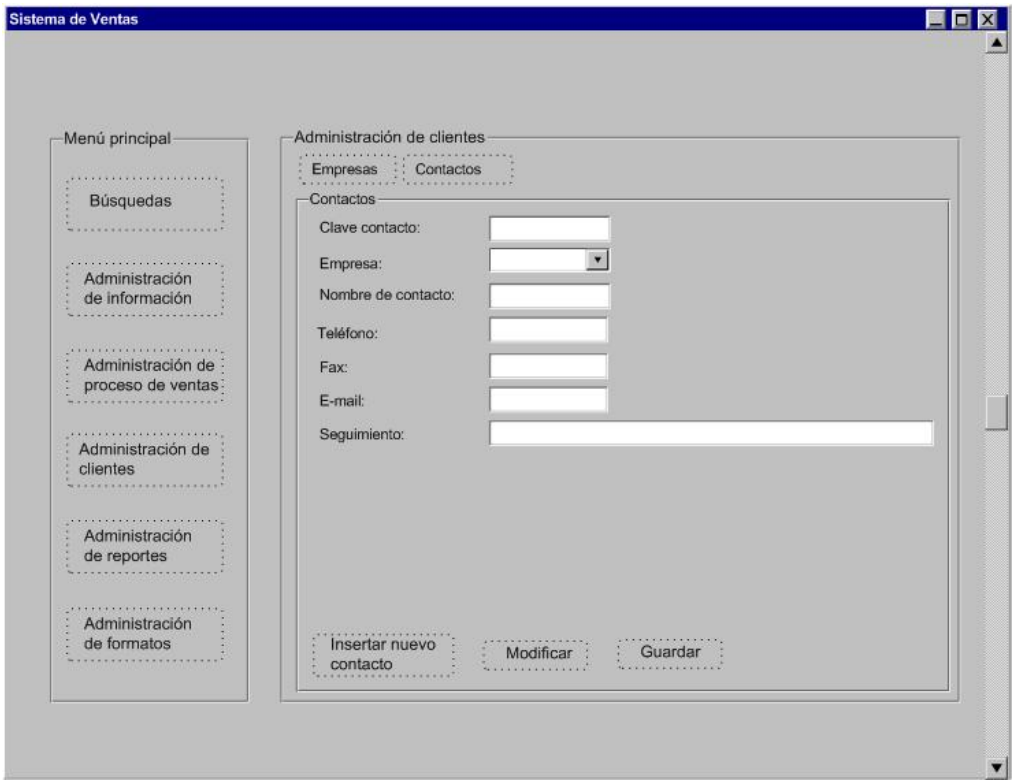


Figura VI.89. Pantalla de administración de clientes por contacto

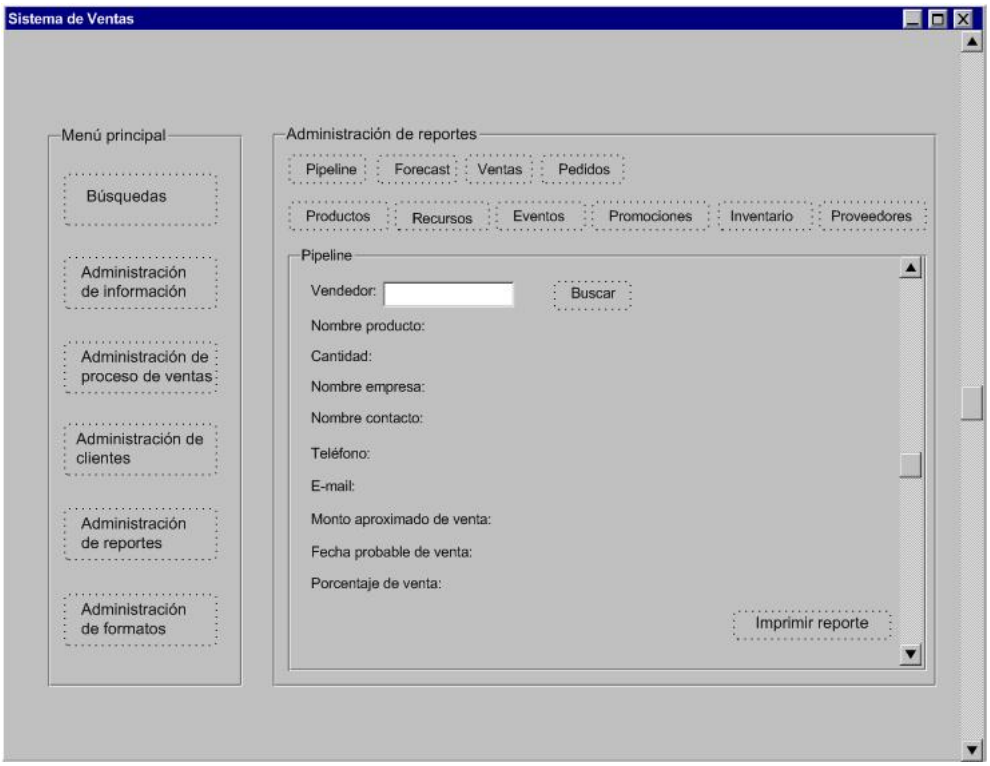


Figura VI.90. Pantalla de administración de reportes de pipeline

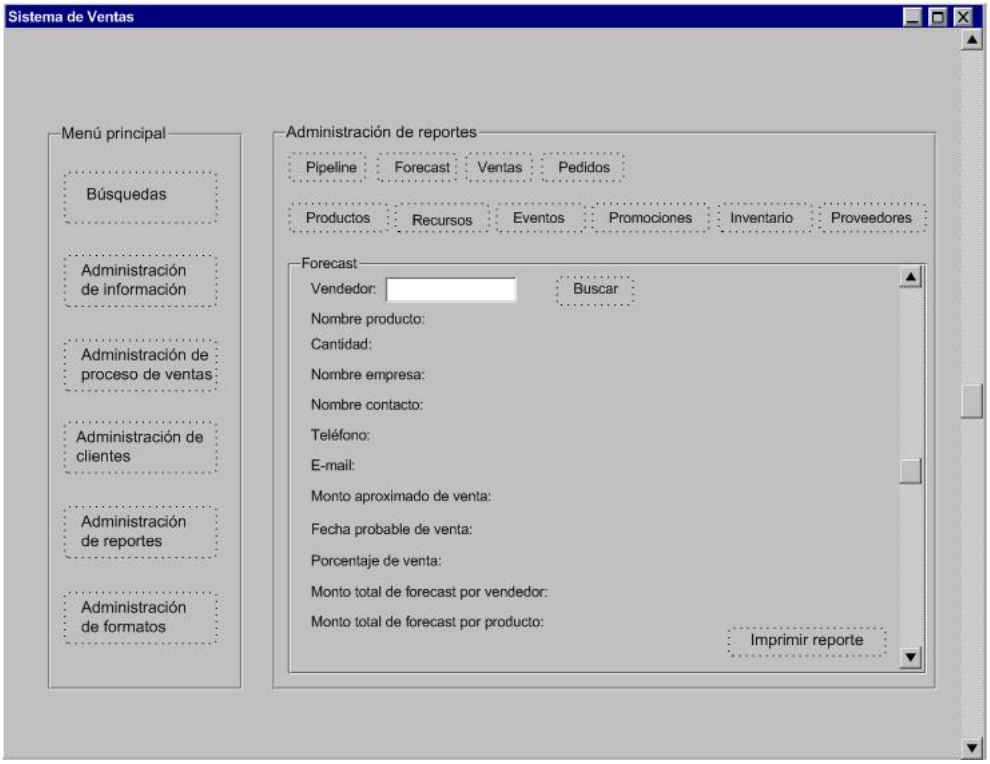


Figura VI.91. Pantalla de administración de reportes de forecast

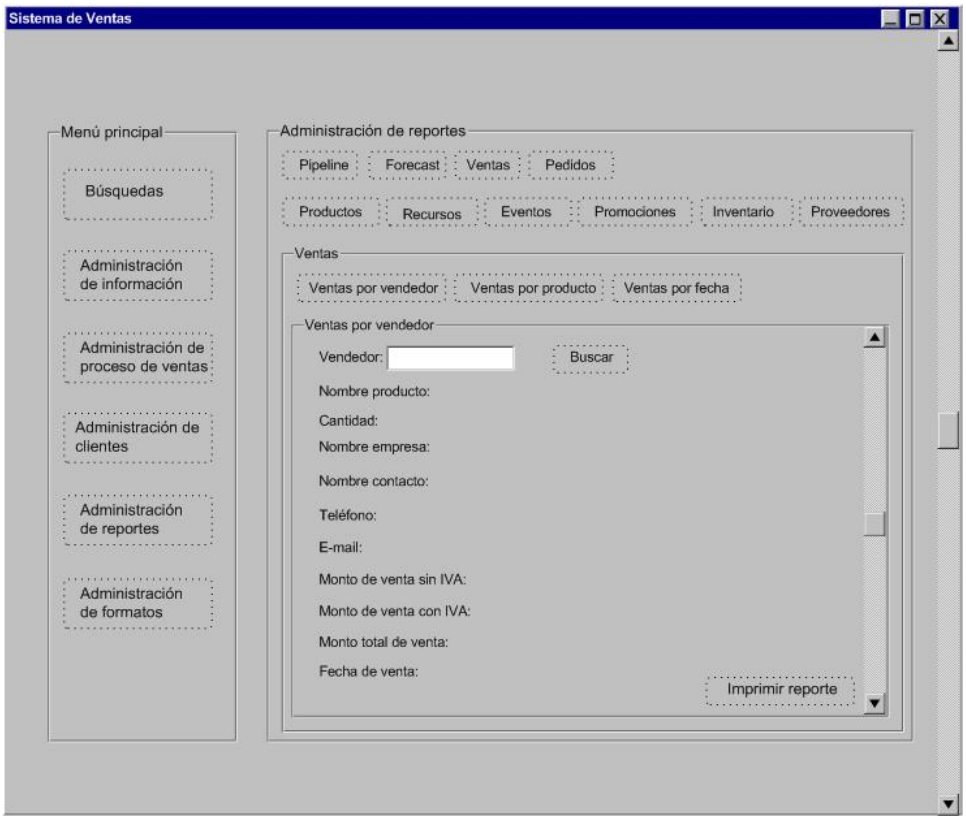


Figura VI.92. Pantalla de administración de reportes de ventas

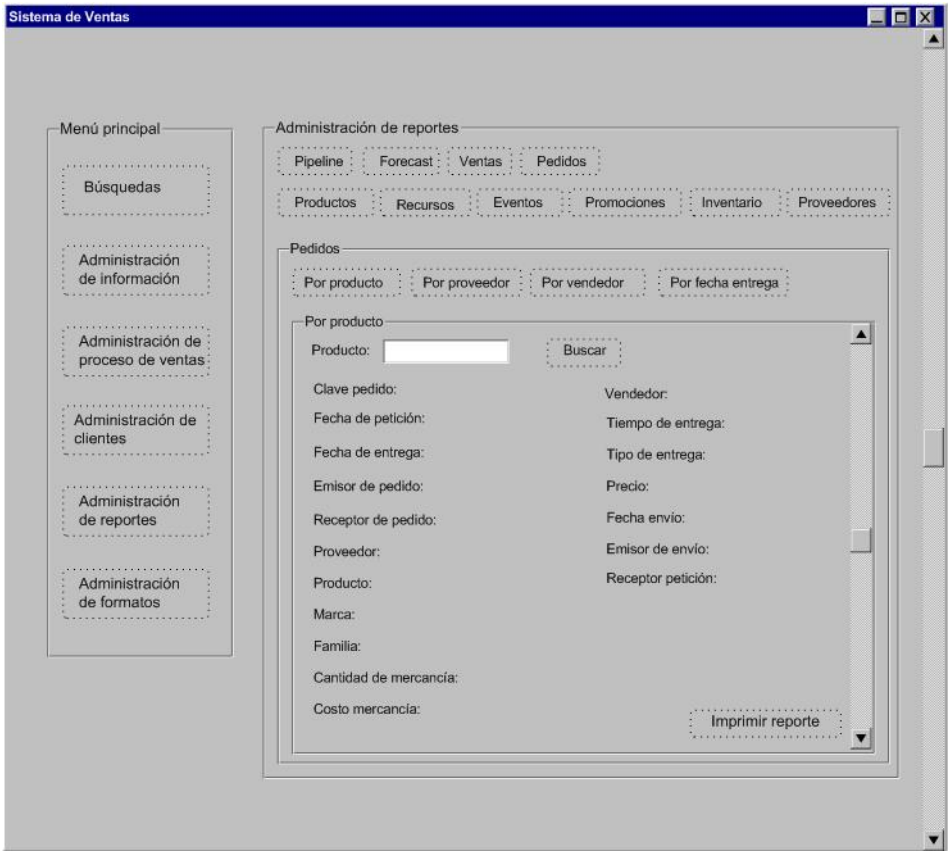


Figura VI.93. Pantalla de administración de reportes de pedidos

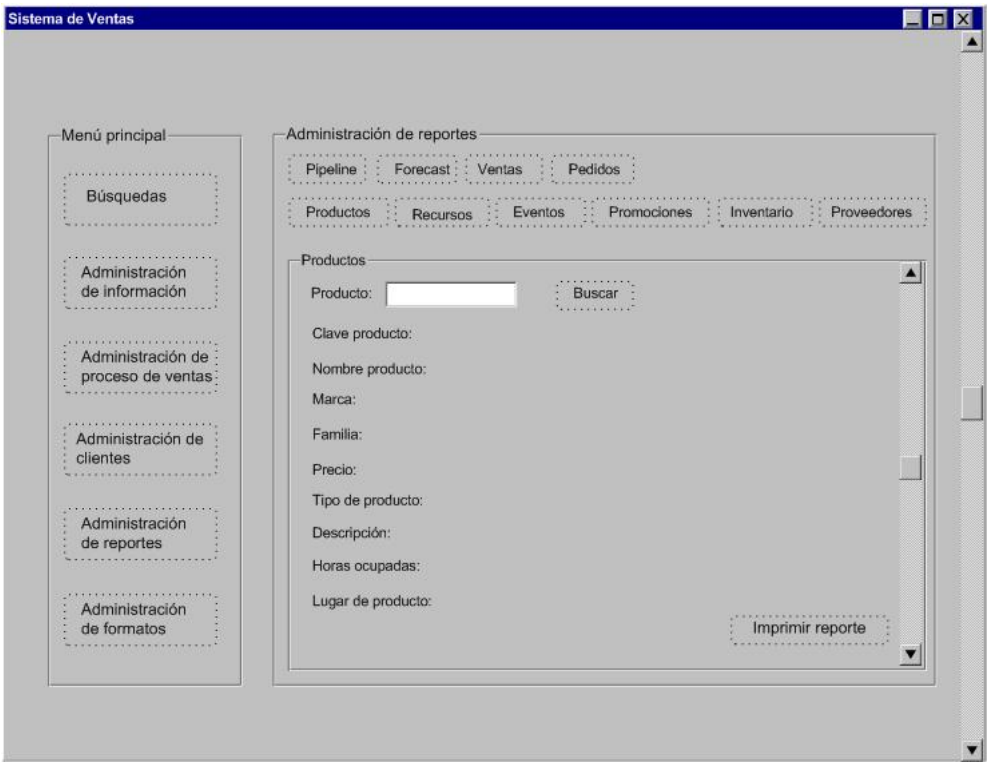


Figura VI.94. Pantalla de administración de reportes de productos

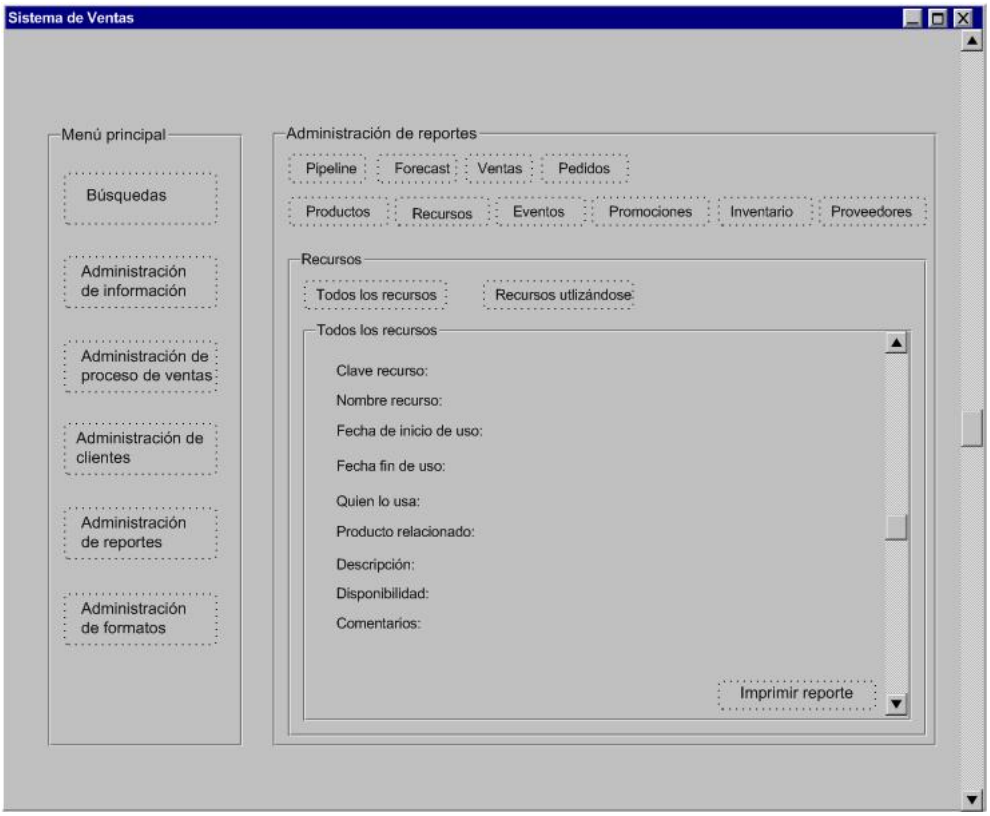


Figura VI.95. Pantalla de administración de reportes de recursos

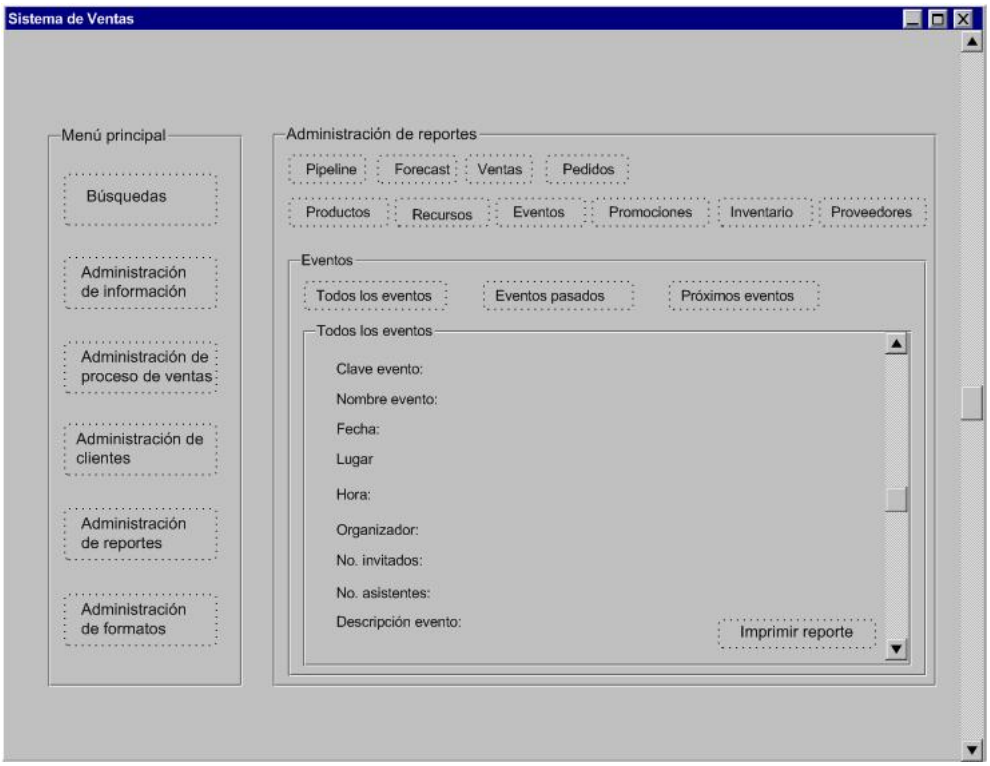


Figura VI.96. Pantalla de administración de reportes de eventos

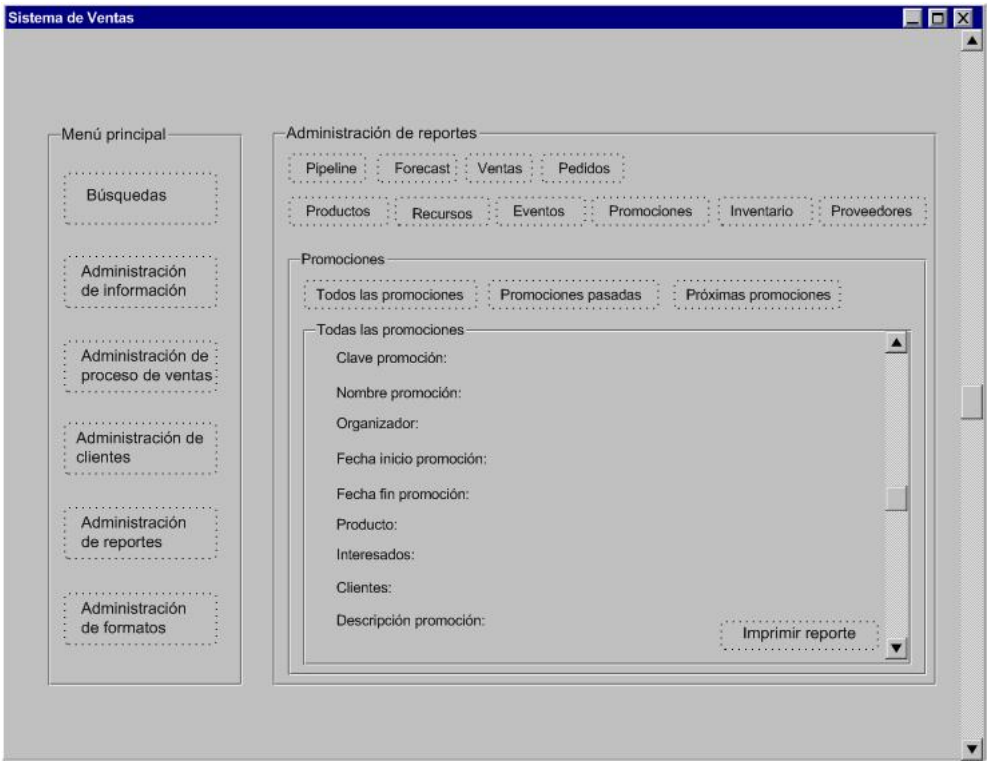


Figura VI.97. Pantalla de administración de reportes de promociones

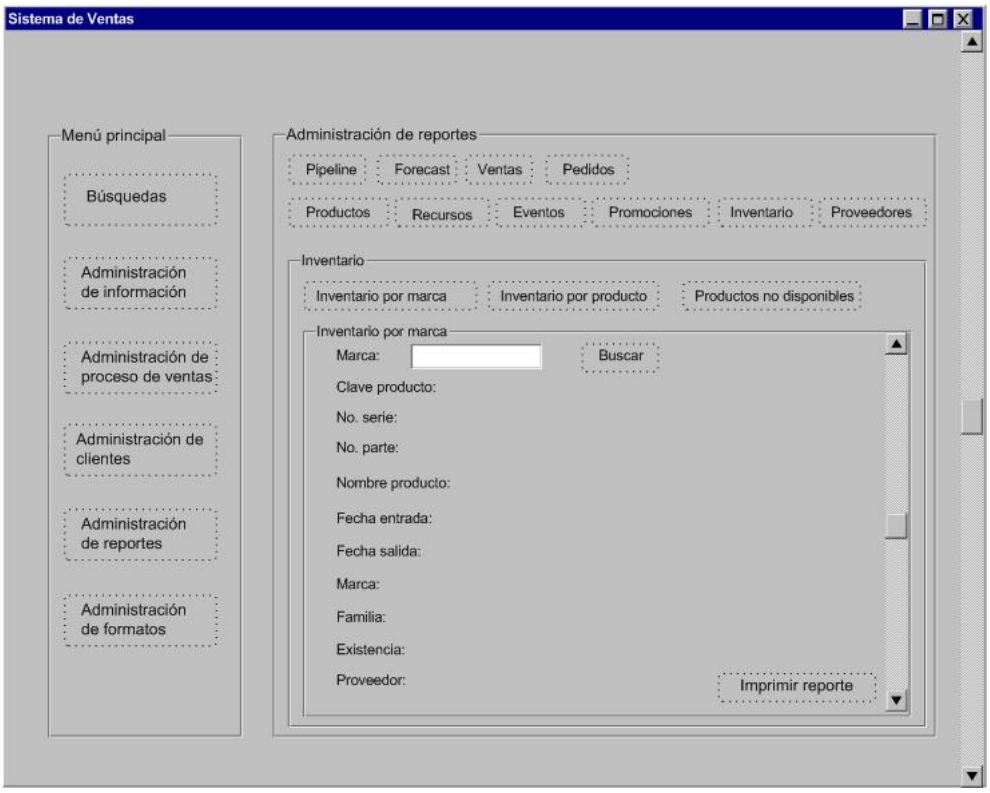


Figura VI.98. Pantalla de administración de reportes de inventario

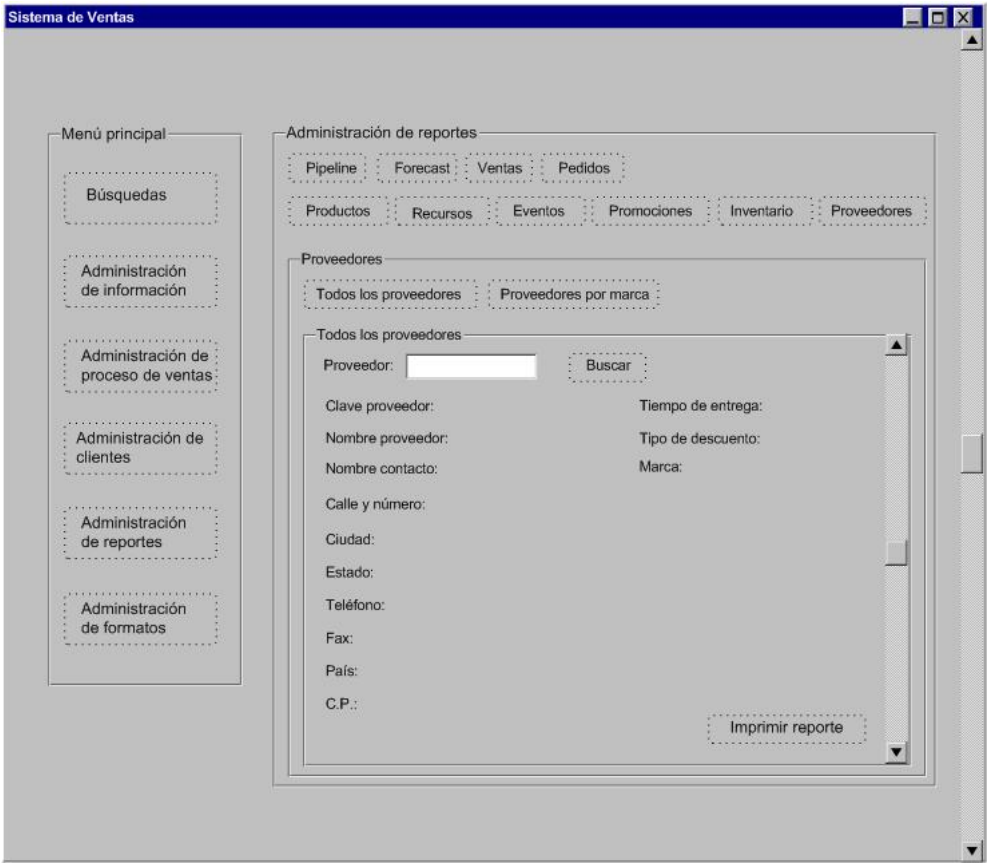


Figura VI.99. Pantalla de administración de reportes de proveedores

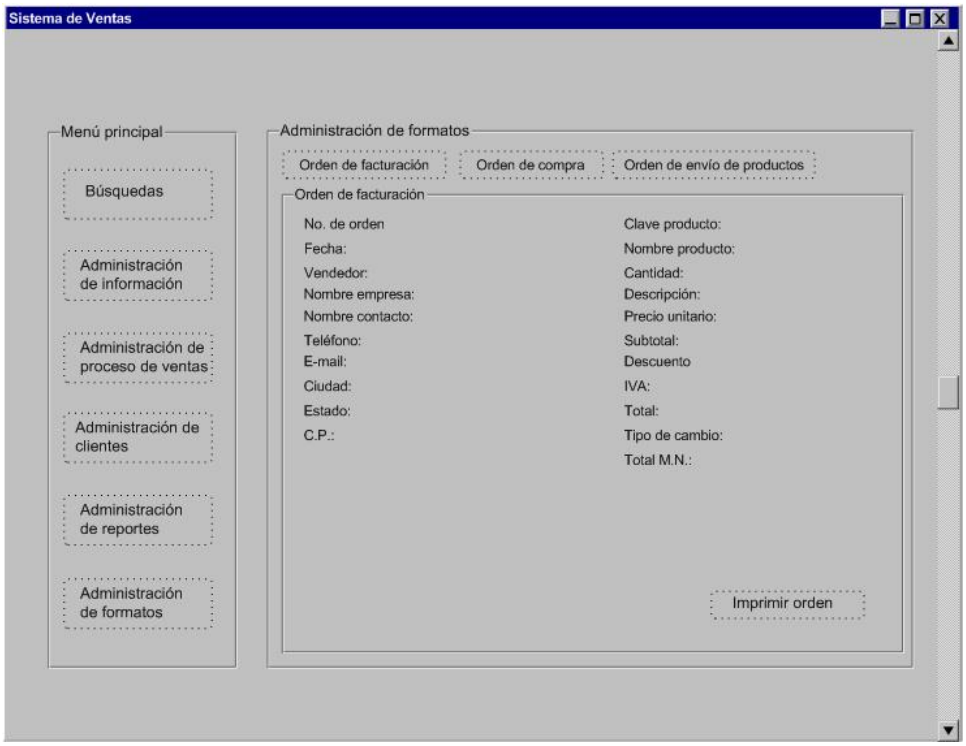


Figura VI.100. Pantalla de administración de formato de orden de facturación

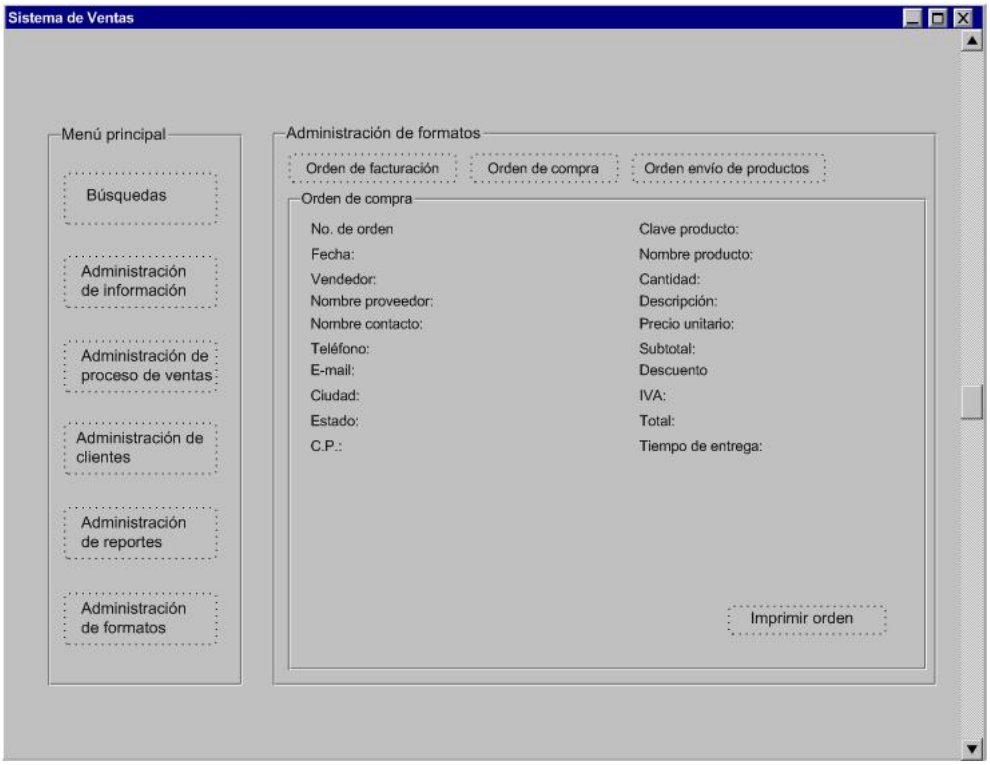


Figura VI.101. Pantalla de administración de formato de orden de compra

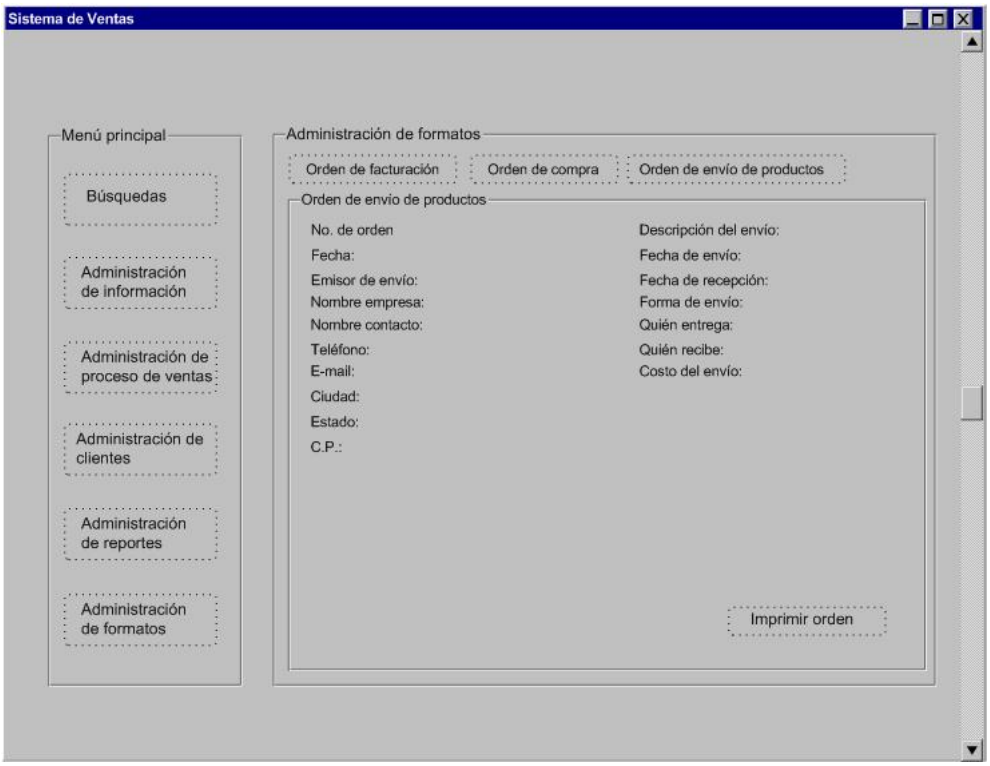


Figura VI.102. Pantalla de administración de formato de orden de envío de productos

En base a las pantallas de la interfaz propuestas anteriormente se pueden identificar los objetos empresariales para el correcto diseño de los procesos, y con ello se definen las siguientes clases.

Dado que este análisis es para el área de ventas, se tiene dentro de este escenario los siguientes objetos, los cuales se muestran como componentes de las clases.

La figura VI.103 nos muestra las clases con sus atributos y operaciones correspondientes para el sistema de ventas, las que posteriormente sirven para el diagrama de clases correspondientes al sistema.

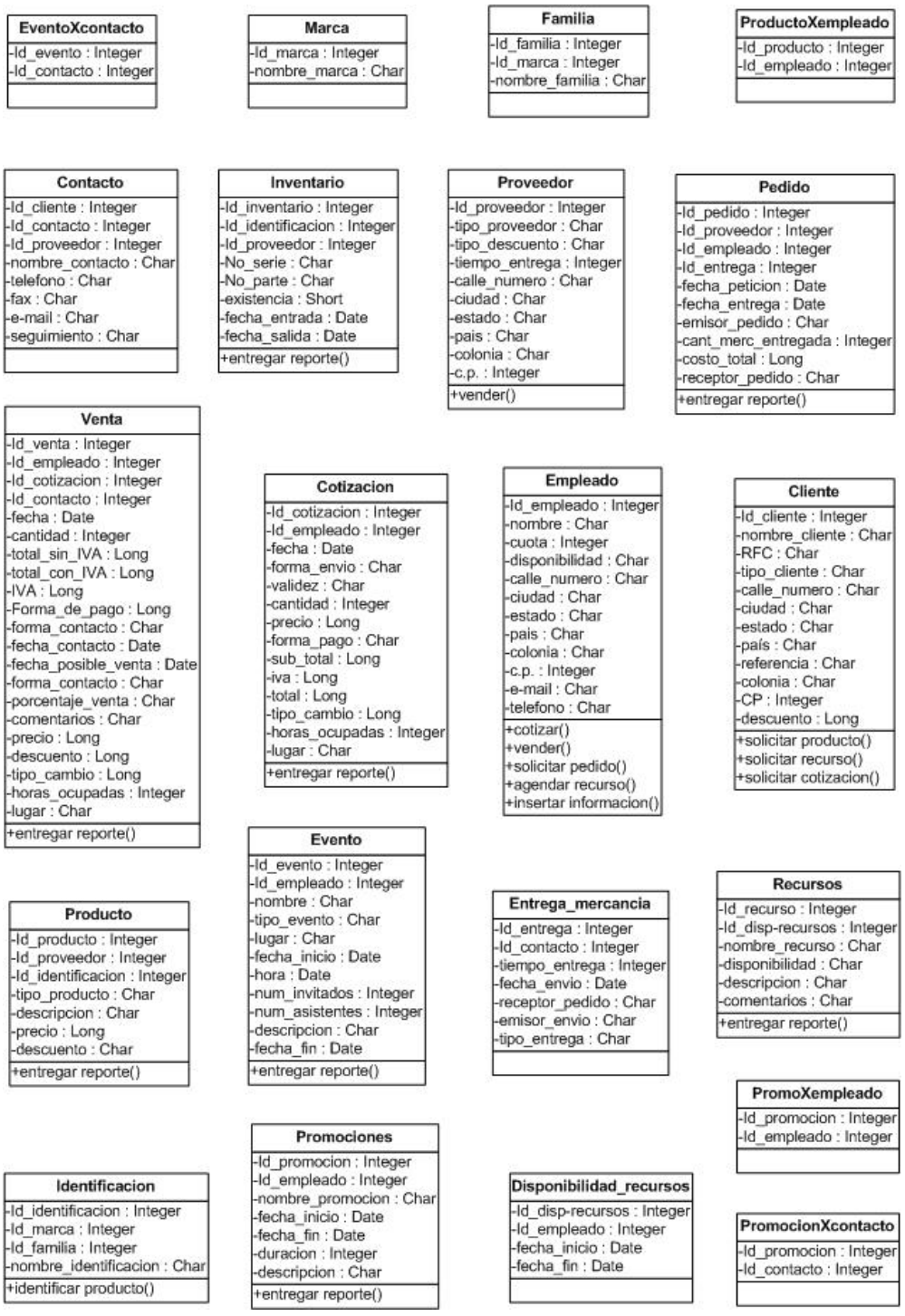


Figura VI.103. Definición de clases para el sistema de ventas

En base a las clases definidas anteriormente, ahora se puede hacer el diagrama de clases, que se muestra en la figura VI.104, el cuál será de ayuda para el análisis y el diseño del sistema, debido a que muestran las propiedades y funcionalidades de los elementos que forman parte del sistema, así como las relaciones que existen entre estos.

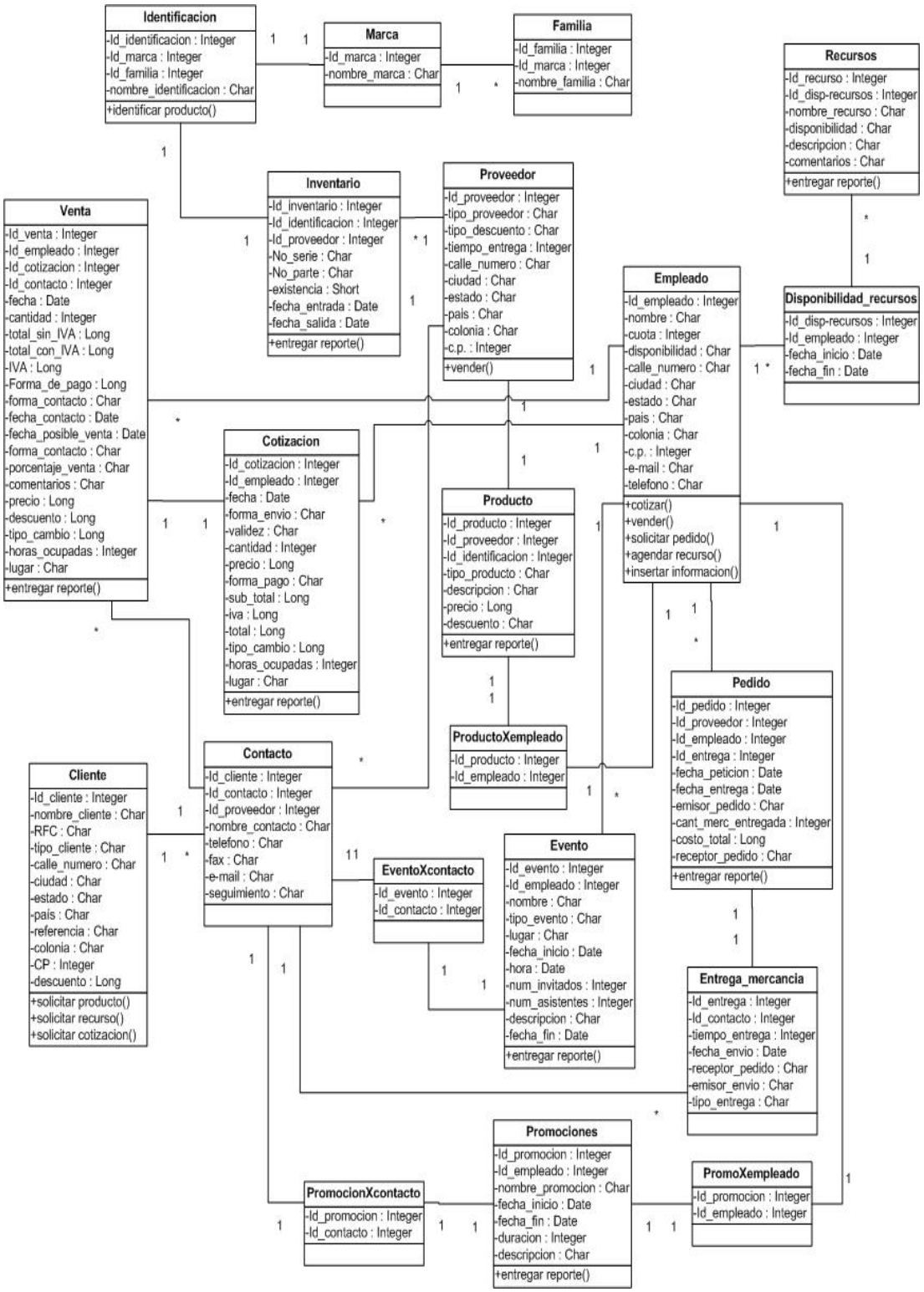


Figura VI.104. Diagrama de clases para el sistema de ventas propuesto

Una vez que se ha mostrado el modelo del diseño del sistema propuesto para mejorar el proceso de ventas, el siguiente aspecto a tratar es el de implementación, el cual incluye la vista de implementación y la vista de despliegue.
A través de estos diagramas se mostrará la arquitectura necesaria para la implementación del sistema propuesto.

En la figura VI.105 se observa el diagrama de implementación por medio de componentes que muestran la forma en que se construirá el sistema, así como las relaciones entre los componentes y sus interfaces. En esta figura se muestra como el cliente hace la petición al CRM, para ello la petición del cliente llega al firewall, este debido a políticas de seguridad va a permitir o negar la petición. Si acepta la petición, el firewall tiene que ubicar donde se encuentra el CRM, para ello consulta al DNS. Una vez ubicado al CRM pasa por el componente de autentificación del cliente, en el cual se decide a que módulos de CRM tiene acceso el cliente. Cuando el cliente está en el CRM se tiene comunicación con la base de datos, la cual recibe y entrega los datos al CRM, para que este los muestre al cliente.

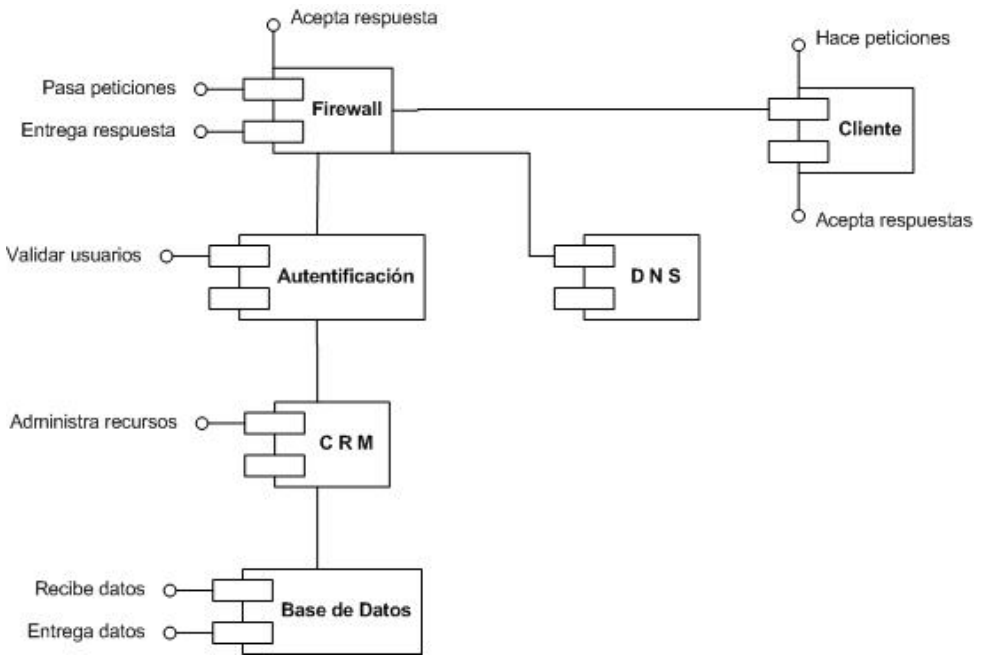


Figura VI.105. Diagrama de implementación para el sistema de ventas

Finalmente el diagrama de despliegue muestra de manera general la distribución de los elementos computacionales necesarios para la implementación del sistema, es decir la ubicación de los componentes en los servidores o hardware, así como los enlaces de redes que se puedan tener.

En la figura VI.106 se muestra el diagrama de despliegue para la implementación del sistema de ventas propuesto. Los elementos que forman el diagrama son: el sistema de ventas (CRM), DNS y Firewall, así como el enlace de red TCP/IP para la comunicación de servidores con el cliente.

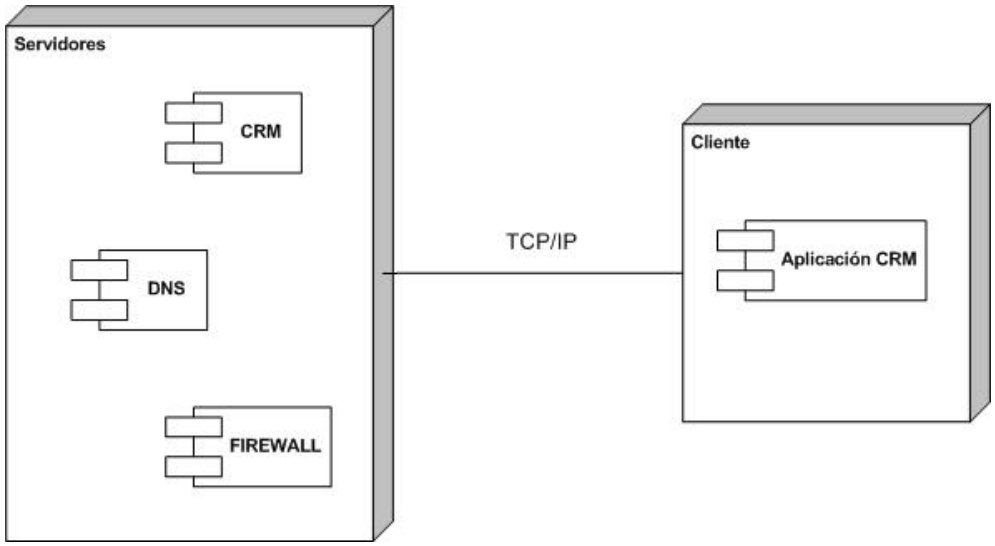


Figura VI.106. Diagrama de despliegue para el sistema de ventas

CONCLUSIONES

CONCLUSIONES

Debido a la gran competencia de negocios que viven las empresas unas con otras, la calidad y la mejora e innovación de procesos se ha vuelto parte esencial, para mejorar el desempeño de los negocios.

La implantación de procesos y la innovación de estos, proporciona beneficios a las empresas que los llevan a cabo, algunos de ellos son: la reducción de tiempo, reducción de costos, competitividad, incremento de número de clientes, mejor coordinación entre los diversos departamentos de la empresa y mayor remuneración financiera para la empresa.

Un proceso orientado a negocios es una estructura que asegura que las actividades que han sido diseñadas para producir una salida específica para un cliente particular son mostradas y dadas a conocer previamente, a todas las personas que están involucradas con el proceso. Esto implica enfocarse en cómo se realiza el trabajo en una organización en lugar de hacer énfasis en el qué es lo que se obtiene.

El presente trabajo proporciona información para el modelado de procesos de negocios, específicamente el proceso de ventas dentro de una empresa, fundamentándose en el modelo UML que permite describir, representar, construir y documentar los elementos de un sistema, entre ellos el de negocios. Los procesos modelados en esta tesis muestran la forma en cómo se realizan las actividades a lo largo del proceso y da ha conocer los elementos que lo forman, las responsabilidades y relaciones de cada uno de los involucrados, proporcionando facilidad de entendimiento del proceso mediante la representación de los diagramas de UML.

La información mostrada permite:

- Analizar los requerimientos del cliente.
- Identificar los objetos que forman parte de la empresa, la manera en como se desenvuelven, los atributos que tienen y las relaciones entre ellos.
- Identificar y evaluar tareas y responsabilidades que el usuario puede realizar.
- Analizar las interacciones entre las diversas actividades y las relaciones que entre ellas.
- Validar los diagramas que describen las secuencias del proceso.
- Analizar la automatización del proceso.
- Diseñar la automatización del proceso.

Los procesos de venta modelados con UML, permiten tener beneficios económicos reducción de tiempo, y satisfacción del cliente con su implantación, por medio de la mejora de atención al cliente, llevando el seguimiento personalizado de cada uno de ellos, dando eficiencia y eficacia y proporcionando la calidad deseada por la empresa.

GLOSARIO

GLOSARIO

Benchmarking. Proceso sistemático y continuo para evaluar los productos, servicios y procesos de trabajo de las organizaciones de negocios, realizado con el propósito de obtener información para ayudar a realizar mejoras organizacionales.

CRM (Customer Relationship Management, Administración de las Relaciones con el Cliente). Son la metodología, la información y los procesos, que permiten a una empresa administrar sus contactos con los clientes de una forma organizada, necesarios para construir una relación entre una compañía y sus clientes.
Sistema de información que tiene como objetivo administrar de forma óptima las relaciones con los clientes.

DNS (Domain Name System, Nombre de Sistema de Dominio). Es un conjunto de protocolos y servicios (base de datos distribuida) que permite a los usuarios utilizar nombres en vez de tener que recordar direcciones IP numéricas.

Estereotipo. El concepto inicial de estereotipo (stereotype) se refería a una clasificación de alto nivel de un objeto que proporciona una idea del tipo de objeto del que se trata. Dicho concepto ha sido tomado en UML para proporcionar un mecanismo de extensión para el propio lenguaje. Este mecanismo hace posible definir UML como un conjunto mínimo de símbolos que podrían ser extendidos, cuando sea necesario, para proporcionar artefactos de modelado que tienen su significado dentro de un proceso de software específico. Mediante el uso de estereotipos se pueden crear nuevos tipos de elementos de modelado basados en los elementos que forman el metamodelo UML, por tanto, un estereotipo será un nuevo tipo de elemento de modelado que extiende la semántica del metamodelo pero no la estructura de los tipos o clases preexistentes.

Estereotipo compile. Define relaciones entre los componentes de un archivo ejecutable.

Estereotipo executable. Define las características de un archivo ejecutable.

Estereotipo file. Define las características de un objeto archivo.

Estereotipo library. Define las características de un objeto librería.

Estereotipo link. Define relaciones entre páginas cliente y otras páginas (cliente o servidoras).

Estereotipo table. Define las características de una tabla que define algún tipo de objeto.

Firewall. Es la combinación de hardware y software la cual separa una red de área local (LAN) en dos o más partes con propósitos de seguridad. Su objetivo básico es asegurar que todas las comunicaciones entre dicha red e Internet se realicen conforme a las políticas de seguridad de la organización que lo instala.

Firmware o Programación en Firme. Es un bloque de instrucciones de programa para propósitos específicos, grabado en una memoria tipo ROM, que establece la lógica de más bajo nivel que controla los circuitos electrónicos de un dispositivo de cualquier tipo. Al estar integrado en la electrónica del dispositivo es en parte hardware, pero también es software, ya que proporciona lógica y se dispone en algún tipo de lenguaje de programación

Forecast. Es la planificación de la demanda. Se utiliza para pronosticar calcular y reportar futuras ventas.

Granularidad. Nivel de modularidad de un sistema. Módulos más pequeños indican una mayor flexibilidad.

Hardware. Conjunto de los componentes que integran la parte material de una computadora.

Instancia. Es una entidad individual con su propio valor e identidad, lo que permite distinguirla de todos los demás objetos. Dados distintos instantes de tiempo durante la ejecución de un sistema, se puede identificar una instancia con esa misma instancia en otros instantes de tiempo, aun cuando cambie el valor de la instancia.

Interfaz. Conexión física y funcional entre dos aparatos o sistemas independientes.

Método Booch. El esquema de Booch desaconseja el uso de diagramas de flujos de datos, herramienta característica de los antiguos métodos y enfatiza un ciclo evolutivo. Define que un sistema es analizado como una serie de escenarios diferentes, en el que cada uno de ellos es descrito por una serie de modelos de diagramas.

Método Fusion. El método de Fusion viene de Hewlett-Packard. Fue llamado de segunda generación de métodos porque es basado en las experiencias de muchos métodos. Heredó una serie de importantes ideas usadas anteriormente, e incluye técnicas para la especificación de las operaciones e intenciones entre los objetos.

Método OOSE/Objectory. Estos métodos fueron construidos sobre el mismo punto de vista básico, creado por I. Jacobson. El método OOSE es una visión de Jacobson de un método orientado a objetos. El método Objectory es usado para construcción de varios sistemas (como de telecomunicaciones para Ericsson, o financieros).

Ambos métodos se basan en el uso de cajas, que definen las características principales del sistema de vistas para el actor externo. Esas cajas son usadas en todas las fases del desarrollo del sistema.

OMT (Object Modeling Technique). Fue creada por James Rumbaugh y Michael Blaha en 1991, mientras James dirigía un equipo de investigación de los laboratorios General Electric. OMT es una de las metodologías de análisis y diseño orientadas a objetos, más maduras y eficientes que existen en la actualidad. La gran virtud que aporta esta metodología es su carácter de abierta (no propietaria), que le permite ser de dominio público y, en consecuencia, sobrevivir con enorme vitalidad. Esto facilita su evolución para acoplarse a todas las necesidades actuales y futuras de la ingeniería de software.

Pipeline. Lista de procesos de negocio, muestra las posibles oportunidades de negocio y cuentas y contactos relacionadas.

Reingeniería. Es la revisión fundamental y el rediseño radical de procesos para alcanzar mejoras espectaculares en medidas críticas y actuales de rendimiento, tales como costos, calidad, servicio y rapidez.

Software. Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

Taxonómica. Ciencia que trata de los principios de clasificación.

TCP/IP (Transmission Control Protocol/Internet Protocol): Sistema de protocolos en los que se basa en buena parte Internet. El primero se encarga de dividir la información en paquetes de origen, para luego recomponerla en el destino. El segundo la dirige adecuadamente a través de la red.

Trazabilidad. Características de las transiciones para saber quién fue el emisor y quien es el receptor.

BIBLIOGRAFIA

BIBLIOGRAFIA

Rumbaugh, James / Jacobson, Ivar / Booch, Grady
El Lenguaje Unificado de Modelado. Manual de Referencia
Ed. Pearson. Addison Wesley, 2000

Rumbaugh, James / Jacobson, Ivar / Booch, Grady
El Lenguaje Unificado de Modelado.
Ed. Pearson. Addison Wesley, 1999

Fowler, Martin / Scott, Kendall
UML Gota a Gota
Ed. Prentice Hall, 1999

Larman, Craig
Análisis y Diseño Orientado a Objetos con UML
Ed. Prentice Hall, 1999

Muller, Pierre-Alain
Modelado de Objetos con UML
Ed. Gestión 2000.

Pressman, Roger
Ingeniería del Software. Un enfoque práctico
Ed. McGraw-Hill, 1998

R. Elmasri, S. B. Navathe
Sistemas de Bases de datos. Conceptos fundamentales
Ed. Addison-Wesley Iberoamericana, 1997

Sliberschatz A., Korth H.
Fundamentos de Bases de Datos
Ed. McGraw-Hill, 1993

Paul, James
Gestión de la Calidad Total
Ed. Prentice Hall, 1997

Davenport, Thomas H.
Process Innovation. Reengineering Work through Information Technology
Publicador. Harvar Business School Press, 1993

REFERENCIAS ELECTRÓNICAS

http://www.sparxsystems.cl/business_process_model.htm

http://www.geocities.com/txmetsb/UML-Use-cases.htm

http://www.bit-net.org/tribuna/tribuna20/interactiva.htm

http://www.microsoft.com/spanish/msdn/articulos/archivo/210901/voices/RAOverview3.asp

http://www.dcc.uchile.cl/~psalinas/uml/modelo.html

http://web.jet.es/amozarrain/reder.htm

http://www.cs.ualberta.ca/~pfiguero/soo/uml/estr_estatica01.html

http://www.fao.org/documents/show_cdr.asp?url_file=/DOCREP/005/X0475S/x0475s0s.htm