Universidad de Castilla-La Mancha

Cuaderno de Prácticas

MODELADO DE APLICACIONES CON UML

PROFESORES

Jose Eduardo Córcoles
María Dolores Lozano

Ingeniería del Software de Gestión

(I.T.I Gestión / I.T.I Sistemas)

TEMA 6. DIAGRAMA DE CLASES. EJERCICIOS RESUELTOS 58

TEMA 7. DIAGRAMAS DE INTERACCIÓN. EJERCICIOS RESUELTOS 82

TEMA 8. DIAGRAMA DE ESTADOS. EJERCICIOS RESUELTOS 89

Tema 1. Introducción a las Herramientas CASE

1.1 Herramientas CASE

CASE es un acrónimo para *Computer-Aided Software Engineering*, aunque existen algunas variaciones para lo que actualmente se entiende por CASE, tal como se ilustra en la Tabla 1.1.

C	Computer
A	Aided Assisted Automated
S	Software Systems
E	Engineering

Tabla 1.1 *Variaciones del acrónimo CASE*

Esencialmente, un CASE es una herramienta que ayuda al ingeniero de software a desarrollar y mantener software. A continuación se presentan algunas definiciones dadas para el término CASE.

En *Terminology for Software Engineering and Computer-aided Software Engineering by B.Terry & D.Logee, Software Engineering Notes, Abril 1990*, CASE es definido como:

> *"Herramientas individuales para ayudar al desarrollador de software o administrador de proyecto durante una o más fases del desarrollo de software (o mantenimiento)."*

En *The CASE Experience, Carma McClure, BYTE Abril 1989 p.235* se ofrece la siguiente definición:

> *"Una combinación de herramientas de software y metodologías de desarrollo"*

La pieza fundamental, y más importante avance tecnológico asociado a una herramienta CASE, es su repositorio integrado. En el repositorio se almacena toda la información de uno o varios sistemas de información, por ejemplo, datos acerca de:

- El dominio (problema) de los sistemas desarrollados o en desarrollo
- Modelos de solución e implementación
- Información de la metodología que está siendo usada
- Historia de los proyectos, recursos, presupuestos, etc.
- Contexto organizacional: organigramas, planes estratégicos, factores críticos de éxito, etc.

Cada ítem en el repositorio es descrito en detalle. Atributos típicos podrían ser: identificación, definición (significado), tipo, alias, ítems componentes, ítems padres, reglas de uso, quién y cuándo lo creó, quién y cuándo lo actualizó por última vez, quiénes pueden actualizarlo y/o consultarlo, cuál es su estado (por ejemplo: incompleto, completo etc.), número de versión, dónde está almacenado físicamente.

1.2 Ingeniería de Software y herramientas CASE

La evolución de las herramientas CASE está ligada a la evolución de la Ingeniería de Software como disciplina. El término "Ingeniería de Software" fue usado por primera vez en una conferencia OTAN en 1968. En dicha conferencia se reveló la existencia de la llamada "Crisis del Software", causada por los problemas inherentes al desarrollo de software.

El ciclo de vida del software es entendido como la secuencia de fases por las cuales atraviesa un proyecto de desarrollo de software desde su concepción hasta el fin del uso del producto software obtenido, pasando por su construcción y mantenimiento. En el Diccionario de la Lengua Española una metodología se define como:

> *"Conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal."*

y un método como:

> *"Procedimiento que se sigue en las ciencias para hallar la verdad y enseñarla."*

Una metodología en *Decline & Fall of the American Programmer, Edward Yourdon, Yourdon Press 1993* es definida como:

> *"Un plan de batalla paso a paso, o libro de cocina, para ejecutar algún resultado deseado. Una metodología de software usualmente identifica las principales actividades- por ejemplo, análisis, diseño, codificación y pruebas- a ser realizadas e indica qué personas (usuarios, administradores, técnicos) deben estar involucrados en cada actividad y qué papel desempeñan en ellas. Las metodologías a menudo describen criterios de entrada (por ejemplo condiciones para comenzar una fase), criterios de salida y puntos de revisión"*

A veces se utiliza ciclo de vida como sinónimo de metodología pues cada metodología ofrece su particular visión de las fases por las cuales debe pasar un proyecto de desarrollo de software.

Un método es entendido como un enfoque técnico para ser utilizado en toda o parte de una metodología. Tanto las metodologías como los métodos están basados en diversas técnicas principalmente gráficas y/o textuales.

Las siguientes son algunas actividades que típicamente incluidas en el ciclo de vida del software: planificación del proyecto, gestión del proyecto, análisis, diseño, codificación, pruebas, documentación, mantenimiento, validación y verificación.

Los esfuerzos iniciales por resolver la "crisis del Software" se orientaron en el ámbito de la codificación, apareciendo las primeras técnicas de programación. A fines de los 60's y comienzos de los 70's surgieron gran cantidad de técnicas para programar y documentar programas. Un ejemplo representativo lo constituyen la Programación Estructurada y los Diagramas de Flujo. Sin embargo, las mejoras introducidas al proceso de desarrollo no eran suficientes.

A fines de los 70's y comienzos de los 80's la atención se centró en resolver problemas de especificación, diseño, métricas y gestión dentro del desarrollo de software. Es decir, el interés fue dirigido a otras fases del desarrollo dentro del ciclo de vida del software.

Actualmente la Ingeniería de Software como disciplina se ve plasmada en una gran variedad de modelos y metodologías para el desarrollo de software. Algunos modelos para el desarrollo de software son: Desarrollo en Cascada, Desarrollo con Prototipación, Desarrollo Incremental, Desarrollo en Espiral. Las metodologías se basan en algún modelo o combinación de ellos.

Actualmente existe un gran número de metodologías tanto comerciales como en el ámbito académico y de investigación. Ellas pueden ser agrupadas en tres grandes corrientes: Metodologías Estructuradas y Metodologías Orientado a Objeto.

1.2.1 Metodologías Estructuradas

Aparecieron a fines de los 60's con la Programación Estructurada, posteriormente a mediados de los 70's extendidas con el Diseño Estructurado y a fines de los 70's con el Análisis Estructurado. Versiones más recientes incorporan Diagramas Entidad-Relación y Diagramas de Transición de Estados.

Ejemplos de metodologías estructuradas promocionadas por organismos gubernamentales lo constituyen: MERISE (Francia), METRICA (España), SSADM (Reino Unido).

Otras metodologías estructuradas en el ámbito comercial son: Gane & Sarson, Ward & Mellor, Yourdon & DeMarco y Information Engineering. Esta última propuesta por James Martin pone un énfasis adicional en el modelamiento de datos y la incorporación de los desarrollos informáticos dentro del contexto su organizacional (planificación, objetivos, etc.).

1.2.2 Metodologías Orientadas a Objeto

Su historia va unida a la evolución de los lenguajes de programación orientada a objeto, los más representativos: a fines de los 60's SIMULA, a fines de los 70's Smalltalk-80, la primera versión de C++ por Bjarne Stroustrup en 1981 y actualmente Java. Sólo a fines de los 80's comenzaron a consolidarse algunas metodologías Orientadas a Objeto.

Algunas de las más representativas en el ámbito comercial son: Booch, Coad & Yourdon, Shaler & Mellor y OMT.

1.3 Evolución de las herramientas CASE

Las primeras herramientas para apoyar el proceso de desarrollo de software fueron los editores y procesadores de texto, usados para escribir programas y su documentación. Así, también algunos programas de dibujo comenzaron a incorporar las notaciones gráficas de técnicas para diseño de programas.

La consolidación de metodologías de desarrollo integrando diferentes técnicas impulsó la aparición de paquetes de propósito más amplio. Surgió la necesidad de un diccionario de datos del sistema que almacene las definiciones usadas en las diferentes fases del desarrollo (este diccionario es lo que actualmente se denomina repositorio). Esto contribuyó a implementar funciones de integración y verificación de consistencia entre técnicas (asociadas a distintas actividades y/o fases en el desarrollo). La automatización de tareas también ha sido un aspecto de interés. En programación automática esto se ha traducido en: generadores de pantallas e

informes, generadores de esquemas físicos de bases de datos y generadores de código para prototipos o partes de programas.

Actualmente, en Ingeniería de Software todos los desafíos y los correspondientes enfoques de solución están siempre concebidos y llevados a la práctica dentro del contexto de un CASE.

1.3.1 Expectativas del uso de un CASE

El propósito de una herramienta CASE es dar soporte automatizado para la aplicación de todas o algunas técnicas usadas por una o varias metodologías.

Cualquier mejora orientada a resolver problemas asociados a la Crisis del Software se enmarca dentro de los siguientes niveles de solución, desde el más general al más particular:

1. Enfrentar el proceso de desarrollo de software como un proyecto de Ingeniería de Software.

2. Aplicar una o varias metodologías de forma integrada cubriendo todas las actividades del ciclo de vida del software

3. Usar una herramienta CASE para apoyar la aplicación de la(s) metodología(s) utilizada(s).

Si consideramos que cada nivel debe implicar al anterior, se pone de manifiesto que la sola utilización de una herramienta CASE no garantiza una mejora en el proceso de desarrollo de software.

Por otra parte, las metodologías incluyen gran cantidad de técnicas, y el esfuerzo de documentación (y actualización de dicha documentación) es por lo general considerable. Por lo tanto, es difícil aplicar una metodología sin la ayuda de una herramienta CASE. Así, los beneficios de utilización de un CASE se entremezclan con los beneficios de aplicar una metodología con éxito. El valor agregado indudable de utilizar un CASE es el aumento en la productividad en las actividades soportadas por la herramienta.

Mientras los costos del hardware han ido en continuo descenso, sucede todo lo contrario con los costos del software. Las exigencias en complejidad y envergadura han sobrepasado a las mejoras en cuanto a métodos de desarrollo. El uso de metodologías de desarrollo junto a herramientas CASE no es una panacea, pero sin lugar a dudas ofrece la mejor alternativa actual para enfrentar proyectos de desarrollo de software de complejidad y/o envergadura.

El énfasis en planificación, análisis y diseño promovido por una herramienta CASE tiene un fuerte impacto y recompensa en la mejora de la calidad del producto obtenido y en el aumento de productividad (disminución de tiempos, costes y esfuerzos) en las actividades de desarrollo y mantenimiento.

El beneficio adicional obtenido por la utilización de un CASE actual (si se compara con la utilización de una metodología sin el uso de un CASE) se representa en los siguientes aspectos:

? Facilita la verificación y mantenimiento de la consistencia de la información del proyecto.

? Facilita el establecimiento de estándares en el procesos de desarrollo y documentación.

? Facilita el mantenimiento del sistema y las actualizaciones de su documentación.

? Facilita la aplicación de las técnicas de una metodología.

? Disponibilidad de funciones automatizados tales como: obtención de prototipos, generación de código, generación de pantallas e informes, generación de diseños físicos de bases de datos, verificadores automáticos de consistencia.

? Facilita la aplicación de técnicas de reutilización y reingeniería.

? Facilita la planificación y gestión del proyecto informático.

Tema 2. Unified Modeling Language (UML)

2.1 Introducción

UML es una notación estándar para desarrollo de sistemas usando el enfoque orientado a objeto. Es una notación en evolución, aún en desarrollo. SA en esta versión provee soporte para la especificación 1.0 de UML.

UML comenzó en 1994 como un esfuerzo de Grady Booch y James Rumbaugh para combinar sus metodologías definiendo una notación estándar para ellas. Después, en 1995, Ivar Jacobson (técnica OOSE , incluyendo Casos de Uso) se unió al equipo.

Paralelamente a este esfuerzo, El *Object Management Group* (OMG, http://www.omg.org) hizo una llamada para propuestas de notación y metamodelo orientado a objetos. En Enero de 1997 UML fue entregado al OMG en respuesta a su convocatoria. Existían otras propuestas remitidas, pero al final, se formó un solo equipo en torno a UML.

UML es sólo una notación, no dicta estándares para el proceso de desarrollo. Sin embargo, UML condiciona dicho proceso de desarrollo al establecer los diagramas e información asociada que debe representarse. Los diagramas incluidos en UML, agrupados según su ámbito de aplicación, son:

Análisis en el contexto organizacional

- Diagramas de Casos de Uso

Análisis y diseño desde la perspectiva estática

- Diagrama de Clase
- Diagrama de Objetos

Análisis y diseño desde la perspectiva dinámica

- Diagrama de Secuencia
- Diagrama de Colaboración
- Diagrama de estados

Implementación

- Diagrama de Componentes
- Diagrama de Despliegue

2.2 Pautas generales para desarrollar usando UML

UML no es una metodología, es una notación obtenida desde experiencia en las más populares metodologías OO actuales. El propósito de este apartado es proveer pautas muy generales de desarrollo, indicando como se relacionan los diagramas de UML.

Existe consenso en que el proceso de desarrollo OO es repetitivo con sucesivos refinamientos. Los diagramas pueden representar situaciones actuales o situaciones propuestas (sistema deseado) en los diagramas de Casos de Uso, de Secuencia o de Colaboración. Al comienzo del proceso de desarrollo, cuando se realiza la captación de requisito, suelen representarse símbolos asociados a objetos (incluso objetos físicos, por ejemplo nombres de personas, cargos, etc.); posteriormente esas definiciones llegan a constituir clases de objetos. Del mismo, modo los mensajes en diagramas de Secuencia y Colaboración dan lugar a los métodos asociados a clases del sistema. Según esto, el nivel de detalle con el cual se especifican los diagramas depende del conocimiento que el desarrollador tiene del sistema. El proceso de aprendizaje del problema y de la determinación de la solución es un proceso de refinamiento sucesivo.

2.2.1 Paquetes y dependencia

En sistemas de envergadura es conveniente hacer una división en subsistemas. En UML esto se representa con el concepto de *package* (paquete). Un paquete es una agrupación de elementos modelados. Los paquetes pueden estar anidados dentro de otros. Todos los tipos de elementos y diagramas pueden ser organizados en paquetes. En particular un paquete puede ser visto como un subsistema el cual es modelado en forma independiente. La relación entre paquetes se establece mediante conexiones de dependencia. Una relación de dependencia se simboliza por una flecha punteada dibujada desde el paquete que al menos usa un elemento de otro paquete, este último es apuntado por la flecha.

El concepto de paquete permite organizar los casos de usos por subsistemas. De la misma forma, en sistemas de tamaño considerable, las clases pueden organizarse en subsistemas.

La Figura 2.1 muestra la representación de paquetes para un ejemplo.

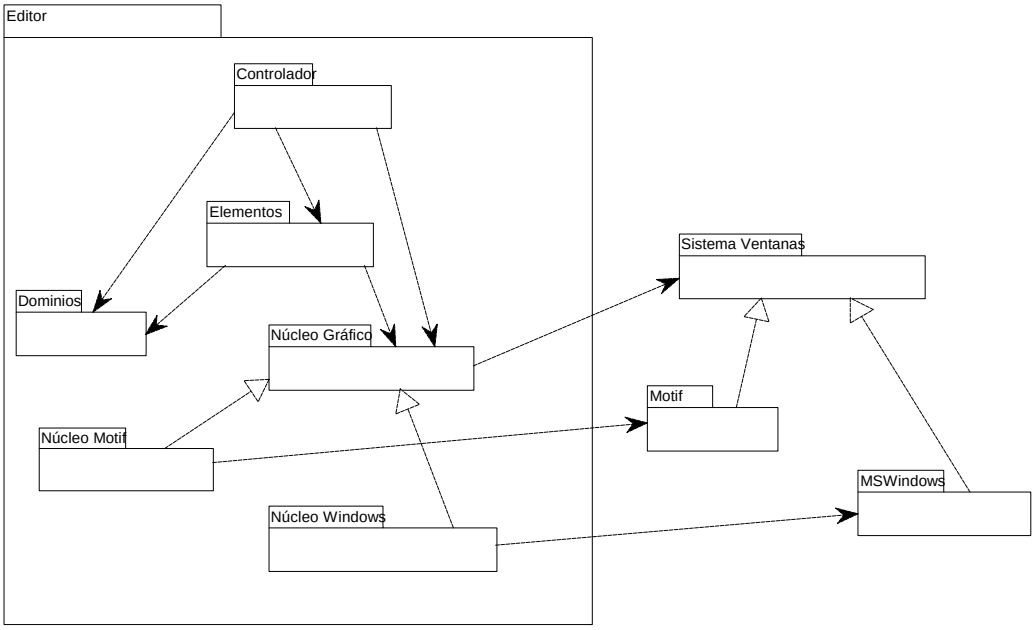


Figura 2.1:*Paquetes.*

2.2.2 Diagrama de Casos de Uso

Casos de Uso es una técnica para capturar información de cómo un sistema o negocio trabaja actualmente, o de cómo se desea que trabaje. No es realmente un enfoque orientado a objeto, más bien es un enfoque de construcción de escenarios en los cuales se modelan los procesos del sistema. Sin embargo, constituye un buen modo de llevar a cabo la fase de captura de requisitos del sistema al comienzo del análisis orientado a objeto.

Típicamente, se modela un Caso de Uso para cada escenario en el sistema o negocio. Cada Caso de Uso puede estar definido simplemente por una sentencia de texto que describe el escenario. También se puede describir mediante una secuencia de pasos ejecutados dentro del escenario o condiciones pre-post para que el escenario comience o termine, respectivamente. Un Caso de Uso es representado por una elipse y describe una situación de uso del sistema interactuando con actores.

Un actor es un agente externo al sistema, alguien o algo que solicita un servicio al sistema o actúa como catalizador para que ocurra algo. UML especifica que un actor es una clase de objetos, no una instancia particular de una clase. Durante el análisis del negocio se puede construir un diagrama de Casos de Uso que represente al sistema y dibujar paquetes que representen los diferentes dominios (subsistemas) del sistema. Para cada paquete se puede crear un diagrama de Casos de Uso hijo donde se describen los casos de uso del dominio. Esto se puede repetir refinando un Caso de Uso en un nuevo diagrama hijo, y así sucesivamente creando una jerarquía de diagramas de Casos de Uso.

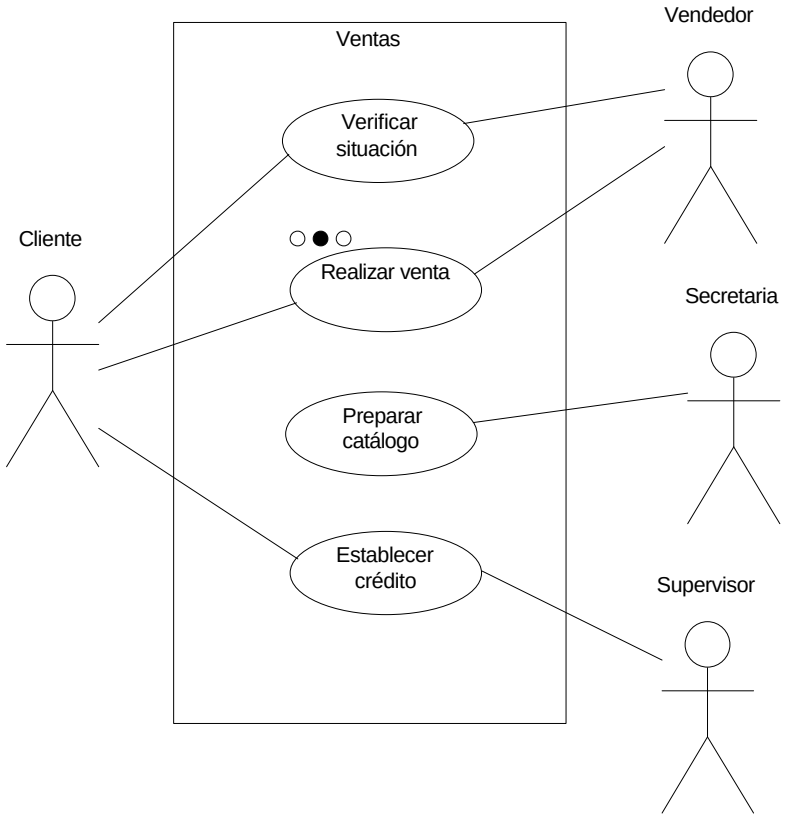


Figura 2.2: *Casos de Uso*

Un diagrama de Casos de Uso es un grafo con actores, un conjunto de Casos de Uso, comunicaciones (participación) entre actores y Casos de Uso, y relaciones entre Casos de Uso. La Figura 2.2 presenta un ejemplo de diagrama de Casos de Uso para un subsistema de ventas.

La *Toolbox* asociada a un diagrama de Casos de Uso se muestra en la Figura de la derecha.

En el último nivel cada Caso de Uso debería ser descrito en mayor detalle, mediante la especificación de pasos asociados, condiciones pre-post o simplemente un texto. La Figura 2.3 presenta la descomposición del caso de uso “emitir orden de venta” de la Figura 2.2.

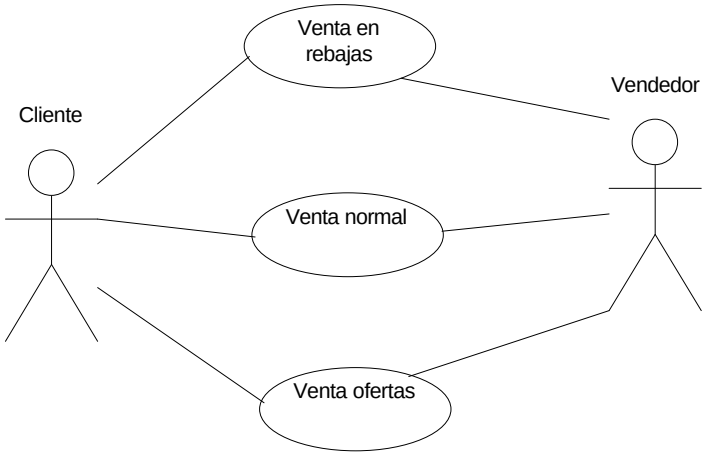


Figura 2.3: ***Casos de Uso Ventas***

En el diagrama de Casos de Uso pueden establecerse tres tipos de relaciones:

a) _ *se comunica con*_: es una relación entre un actor y un caso de uso.

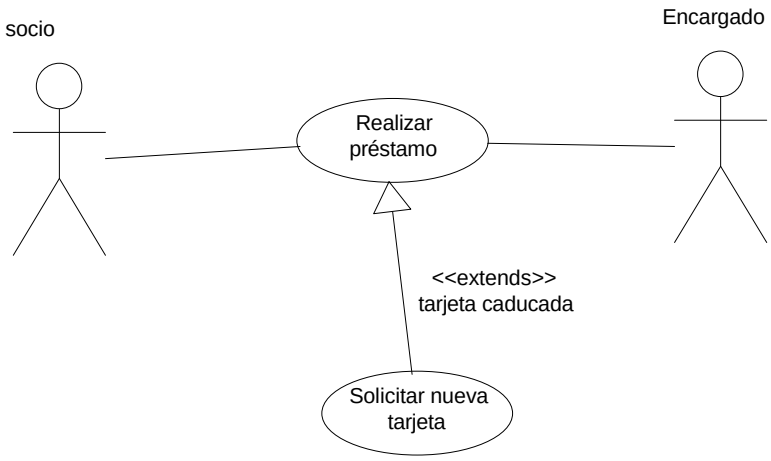


Figura 2.4: *Casos de Uso Prestamo.*

b) _ *extiende*_: es una relación entre un caso de uso y otro. Si al caso de uso A *extiende* al caso de uso B, significa que una instancia del caso de uso B puede incluir (sujeto a ciertas condiciones especificadas en la extensión) el comportamiento descrito en el caso de uso A. Es decir, los pasos del caso de uso A son una extensión a los del caso de uso B. Después de realizados los pasos de A se continúa en el paso siguiente en B. La Figura 2.4 muestra un Caso de Uso “solicitar nueva tarjeta”, que extiende al Caso de Uso “préstamo de libros”.

c) _ *usa* _: es una relación entre casos de uso que indica que un caso de uso “usa” funciones provistas por otro. La relación se representa por una línea desde el caso de uso que utiliza los servicios de otro. Si el caso de uso A *usa* el caso de uso B,

significa que una instancia del caso de uso A incluye también parte del comportamiento especificado por B. En el ejemplo de la Figura 2.5 los casos de uso "Reintegro desde cuenta corriente" y "Reintegro desde cuenta de ahorro" ***usan*** el caso de uso "Verificar Saldo".

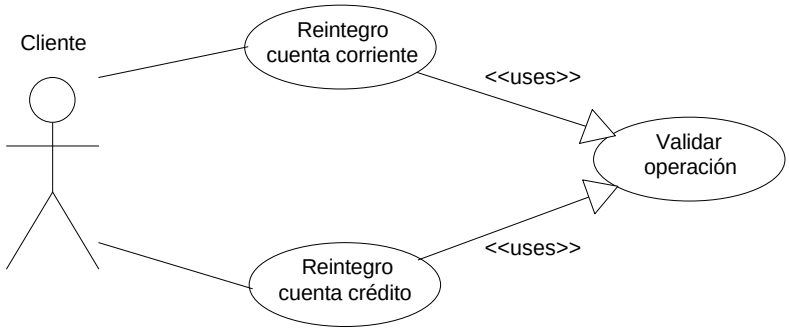


Figura 2.5: *Casos de Uso Reintegro.*

Las relaciones usa y extiende permiten evitar redundancias en la descripción de los casos de uso.

2.2.3 Diagrama de Secuencia y diagrama de Colaboración

El diagrama de Secuencia y el diagrama de Colaboración son realizados al mismo tiempo por Rational Rose y se mantienen sincronizadas sus modificaciones. Al crear/abrir uno de ellos, automáticamente se crea/abre el otro.

Los diagramas de Secuencia y de Colaboración son usados para establecer mayor detalle de un escenario del sistema, determinando los objetos y mensajes involucrados.

El diagrama de Secuencia muestra los objetos involucrados en el escenario mediante líneas verticales y punteadas, y los mensajes entre objetos como flechas horizontales conectando líneas de pares de objetos. Los mensajes son dibujados cronológicamente desde arriba hacia abajo. La ubicación de los objetos es arbitraria. La Figura de la derecha muestra la *Toolbox* asociada a un diagrama de Secuencia.

La Figura 2.6 muestra un diagrama de Secuencia para el escenario "préstamo de libros".

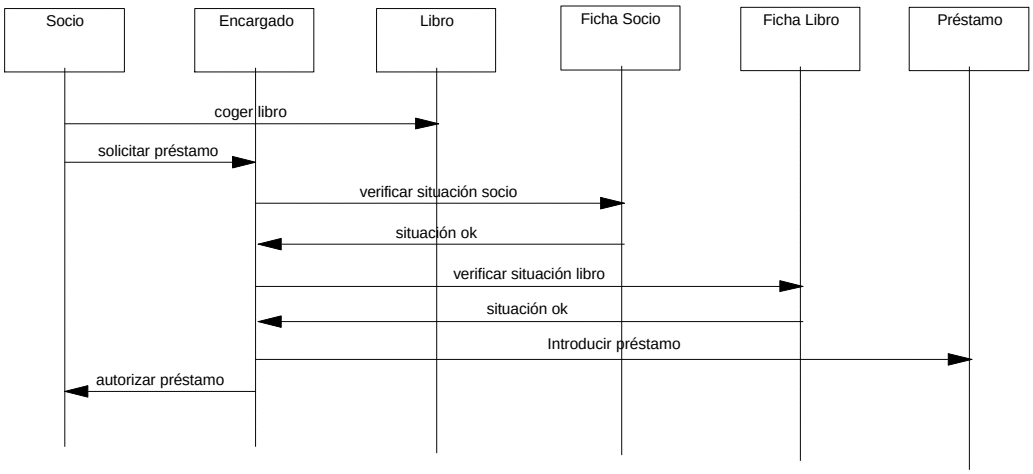


Figura 2.6: ***Diagrama de Secuencia*** *Prestamo de Libros*

El diagrama de Colaboración es usado para modelar la interacción entre los objetos de un Caso de Uso. Los objetos están conectados por enlaces (*links*) en los cuales se representan los mensajes enviados acompañados de una flecha que indica su dirección. El diagrama de Colaboración ofrece una mejor visión del escenario cuando el analista está intentando comprender la participación de un objeto en el sistema. La Figura de la derecha muestra la *Toolbox* asociada a un diagrama de Colaboración.

La Figura 2.7 muestra el diagrama de Colaboración correspondiente al diagrama de Secuencia de la Figura 2.6.

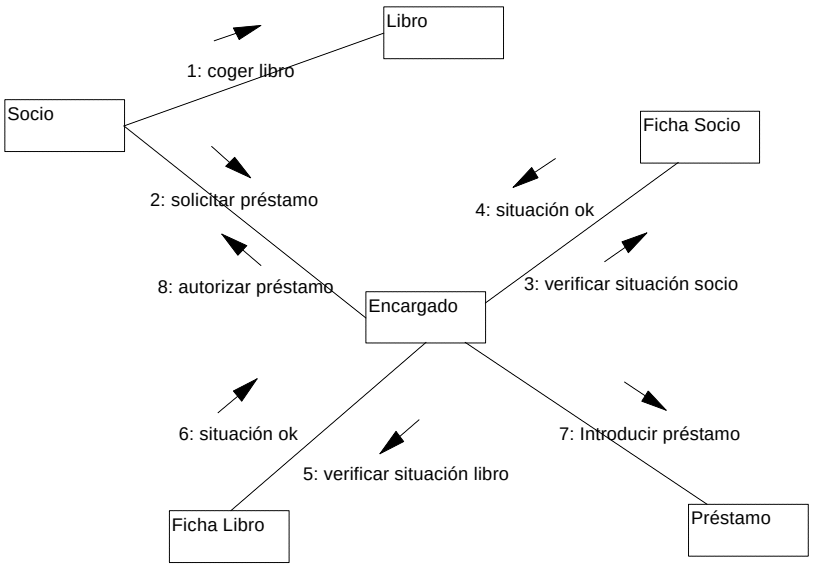


Figura 2.7: *Diagrama de Colaboración*

Inicialmente el análisis pone nombre a los mensajes utilizando el nombre usado en el contexto del negocio. Más tarde, durante el diseño, los nombres de mensajes corresponden a métodos invocados desde un objeto a otro. El método invocado corresponde a un método definido para el objeto al cual apunta la flecha del mensaje.

2.2.4 Diagrama de Objetos y diagrama de Clases

La notación de ambos tipos de diagramas es similar. La diferencia es el nivel de abstracción en el cual se trabaja. Comúnmente, al principio del desarrollo, cuando aún no está decidido qué elementos llegarán a ser clases, objetos o datos elementales, se puede modelar dichos elementos como objetos (instancias de alguna clase). Sin embargo, cuando se establece la solución, todos los elementos están claramente diferenciados y representados en diagramas de Clases (que también podrían incluir representaciones para objetos, cuando sólo una instancia participa en el sistema). Por lo anterior, para el soporte de UML, Rational Rose proporciona sólo un tipo de diagrama, denominado *Class Diagram*.

El diagrama de Clases es el diagrama principal para el análisis y diseño estático. Un diagrama de clases presenta las clases y objetos del sistema con sus relaciones estructurales y de herencia. La definición de clase u objeto incluye definiciones para atributos y métodos.

El trabajo realizado y expresado en los diagramas de Casos de Uso, diagramas de Secuencia y diagramas de Colaboración debería aportar información para la determinación de las clases, objetos, atributos y métodos, es decir, la captura de requisitos. El diagrama de Clases debería permitir la especificación en un ámbito de detalle igual o mayor al utilizado para especificar los tres diagramas anteriores. El diagrama de Clases es la especificación de requisitos en su aspecto estático.

La Figura 2.8 presenta un diagrama de Clases para un sistema de Línea Aérea.

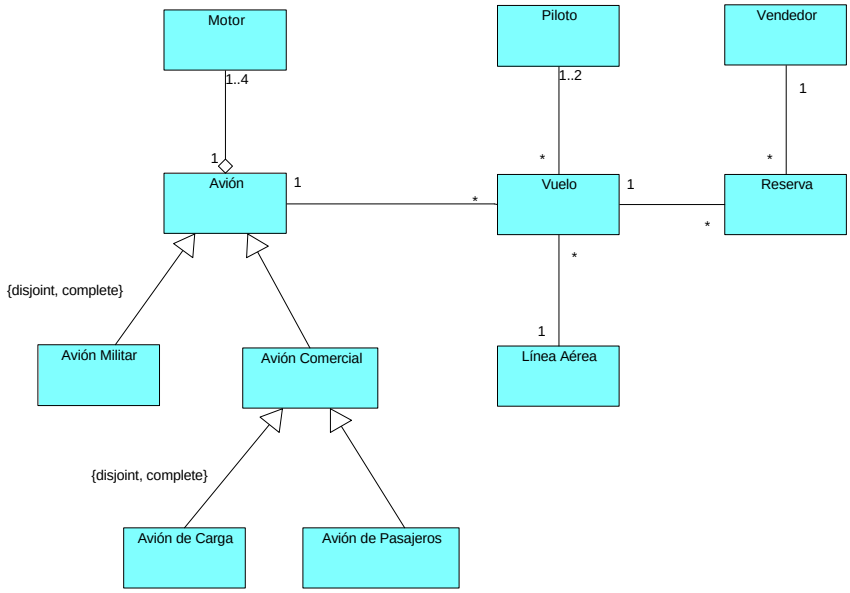
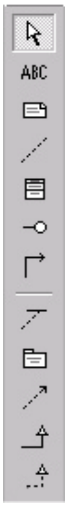


Figura 2.8: *Diagrama de Clases Línea Aerea*



La *Toolbox* usada para construir un diagrama de Clases se muestra en la Figura de la derecha

A continuación se presentan ejemplos de cómo pueden combinarse los elementos gráficos del diagrama de Clases, explicando brevemente los cuadros de diálogo asociados.

Clases

Una clase describe un conjunto de objetos con estructura y comportamiento similares. Una clase está definida por una serie de propiedades. Las más relevantes son los atributos y los métodos. Una clase es representada como un rectángulo con cuatro compartimentos separados por líneas horizontales. En el superior, se muestra el nombre de la clase y todas las propiedades generales, en el siguiente se presenta la lista de atributos de la clase, a continuación se muestran los métodos.

Cada atributo o método de una clase es antecedido por un indicador de visibilidad, especificado en cuadro de diálogo que define al método o atributo

Asociaciones

Son relaciones bidireccionales potenciales entre instancias de clases. En una asociación, una instancia de una clase puede estar relacionada (potencialmente) con un número mínimo y máximo instancias de otra clase (o de la misma clase). Estos números definidos desde la perspectiva de cada una de las clases que están asociadas se denominan cardinalidad de la asociación. La notación para cardinalidad de las asociaciones y relaciones en diagramas de Clase UML es:

Uno y sólo uno: 1

Cero o uno:	0..1
Uno o más:	1..*
Cero o más:	0..*
Entre N y M :	N..M , donde N y M son números naturales y M>=N.

Generalización

La generalización/especialización son operadores entre clases que permiten definir nuevas clases mediante el mecanismo de herencia. La herencia permite separar la especificación de una clase en versiones más refinadas de ella misma. Las versiones refinadas son llamadas subclases o clases derivadas. La clase que es refinada se denomina superclase o clase base. Las subclases están conectadas a las superclases mediante el símbolo "hereda desde". Mediante generalización/especialización se construyen jerarquías de clases. La Figura 2.9 muestra un ejemplo de una jerarquía de generalización/especialización. En dicha figura, Empleado es una generalización de Administrativo, Ejecutivo y Operario. También podemos decir que tanto Administrativo como Ejecutivo y Operario son especializaciones de Empleado. En ambas visiones (generalización o especialización), las subclases Administrativo, Ejecutivo y Operario heredan las propiedades de la superclase Empleado.

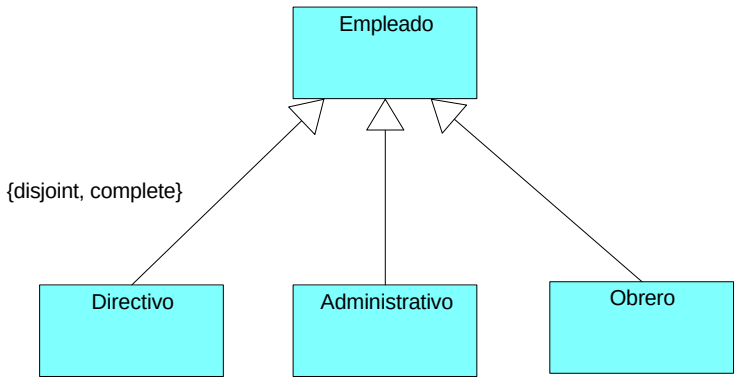


Figura 2.9: *Generalización*

Una clase puede ser superclase de más de una jerarquía. Para cada jerarquía puede especificarse un discriminador, es decir, un nombre que identifica a la jerarquía. Cada jerarquía o cada conexión de subclase puede estar etiquetada con alguna restricción a cumplir por las instancias de la subclase.

Las siguientes son las restricciones predefinidas usadas para jerarquías:

overlapping	Una objeto puede pertenecer a más de una de las subclases.
disjoint	Un objeto puede pertenecer a lo más a una de las subclases.

complete La unión de los objetos instancias de las subclases corresponde exactamente a los objetos instancia de la superclase.

incomplete Un objeto instancia de la superclase puede no pertenecer a ninguna de las subclases declaradas.

2.2.5 Diagrama de Estados

El diagrama de estados representa el comportamiento del sistema a través del tiempo. Típicamente se elabora un diagrama de Estados para cada clase que tenga un comportamiento significativo. El comportamiento es modelado en términos del estado en el cual se encuentra el objeto, qué acciones se ejecutan en cada estado y cuál es el estado al que transita después de un determinado evento.

Los estados representan condiciones que son válidas en el objeto en un cierto instante. Los eventos representan la causa que origina el cambio desde un estado a otro y están asociados a transiciones, líneas que unen el estado inicial y el final. Las acciones ocurren cuando un objeto llega a un estado. La Figura 2.10 muestra la clase socio. A dicho símbolo se le ha enlazado un diagrama hijo de tipo diagrama de Estado (por esta razón aparecen los puntos en el costado superior izquierdo). La Figura 2.11 muestra dicho diagrama de Estados.

La Figura de la derecha muestra la *Toolbox* usada en el diagrama de Estados.

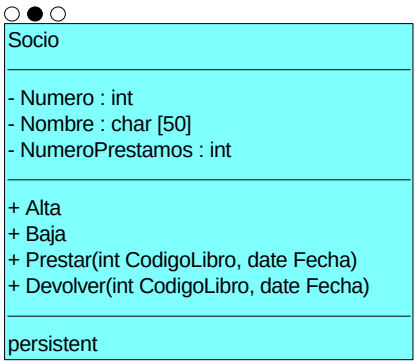


Figura 2.10: *Clase Socio*

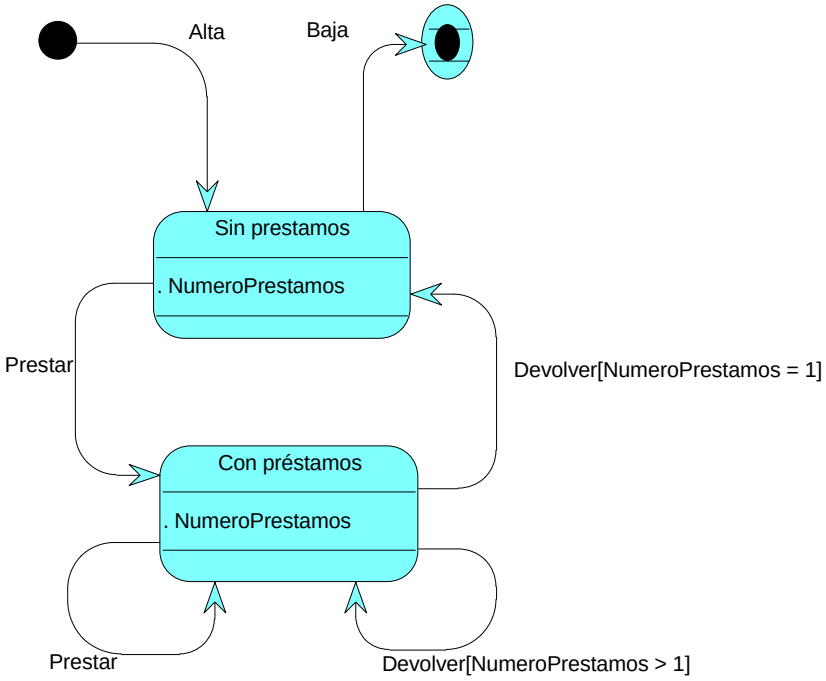


Figura 2.11: *Diagrama de Estados Libro*

2.2.6 Diagrama de Componentes

Un diagrama de Componentes permite modelar la estructura del software, incluyendo dependencias entre componentes en código fuente, componentes en código binario y componentes ejecutables. El diagrama de componentes establece relaciones de dependencia entre componentes y/o paquetes de componentes. La Figura 2.12 muestra un diagrama de Componentes.

Un componente es un grupo de clases que trabajan estrechamente. Los componentes pueden clasificarse por tipos. Algunos sólo existen en tiempo de compilación; otros sólo en tiempo de enlace; otros en tiempo de ejecución y otros en varios de esos momentos.

Una dependencia indica que un elemento del modelo (fuente) depende de otro (objetivo), de manera que un cambio en el elemento objetivo puede significar cambiar el elemento fuente.

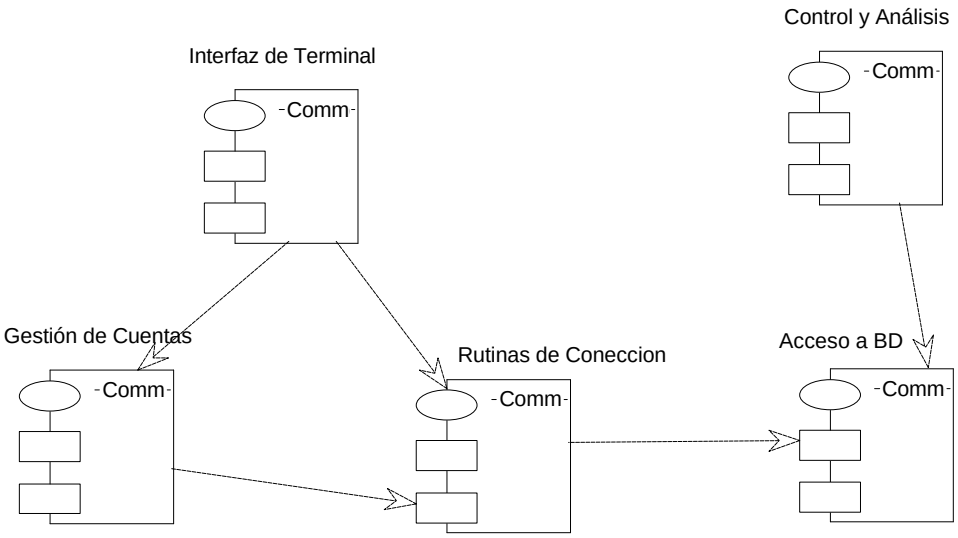


Figura 2.12: *Diagrama de Componentes*

2.2.7 Diagrama de Despliegue

El diagrama de Despliegue modela la distribución en tiempo de ejecución de los elementos de procesamiento y componentes de software, procesos y objetos asociados. En el diagrama de Despliegue se modelan los nodos y la comunicación entre ellos. Cada nodo puede contener instancias de componentes. La Figura 2.13 presenta un ejemplo de diagrama de Despliegue.

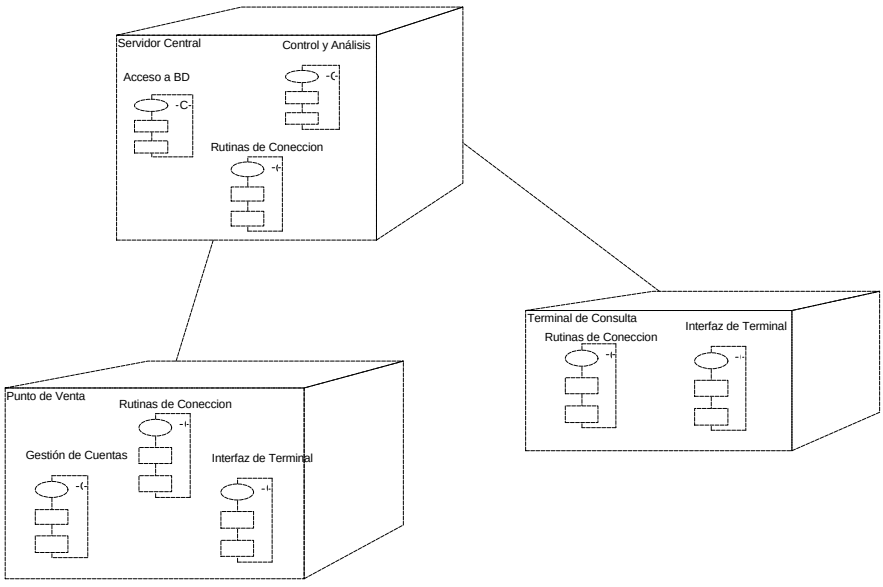


Figura 2.13: *Diagrama de Despliegue*

Tema 3. Diagramas de UML en Rational Rose. Guía de Prácticas

3.1 Introducción

En este tema se detallan una serie de actividades que sirven como práctica inicial para el manejo de Rational Rose. El objetivo fundamental es familiarizarse con el entorno de trabajo al mismo tiempo que se empieza a tomar un primer contacto con la sintaxis y semántica de los diagramas UML.

Nuestro agradecimiento a Patricio.Letelier (www.dsic.upv.es/~uml), profesor de la Universidad Politécnica de Valencia por compartir este trabajo.

3.2 Actividad 1

Con el botón derecho del ratón y estando en el navegador sobre el paquete de la Vista de Casos de Uso, haga **new-package** y cree un paquete que se llame **Actividad 1**.

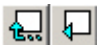
Estando sobre el paquete recién creado haga click con el botón derecho y cree dos nuevos paquetes que se llaman **Ventanas** y **Editor**, estos se crearán como paquetes dentro del paquete **Actividad 1**.

Repita la operación anterior y cree los subpaquetes **Motif** y **MSWindows** como subpaquetes de **Ventanas** y **Controlador**, **Dominio**, **Elementos**, **Núcleo Motif**, **Núcleo Windows** como subpaquetes de **Editor**.

Sobre el paquete **Actividad 1** realice **new-Use Case Diagram**, creando el diagrama **Actividad 1**. Haga doble click en el icono del diagrama e introduzca el diagrama mostrado en la Figura 3.1. Para ello arrastre desde el navegador los paquetes involucrados.

Repita el paso anterior para los paquetes **Ventanas** y **Editor** obteniendo los diagramas mostrados en las Figuras 3.2 y 3.3, respectivamente. En cada oportunidad arrastre desde el navegador los paquetes indicados.

Consejo: Cuando quiera asociar un nuevo diagrama a un paquete basta con hacer doble clic sobre él y luego renombrar el diagrama obtenido (por defecto se denomina **Main**).

Consejo: Utilice los botones  para ir al diagrama padre o al diagrama anterior, respectivamente.

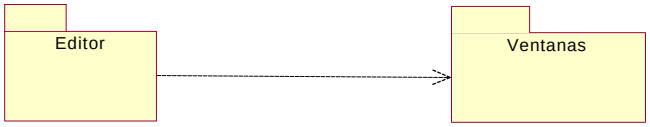


Figura 3.1: *Diagrama Actividad 1*

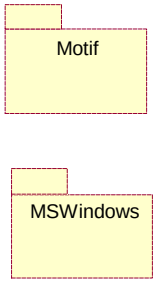


Figura 3.2: *Diagrama Ventanas*

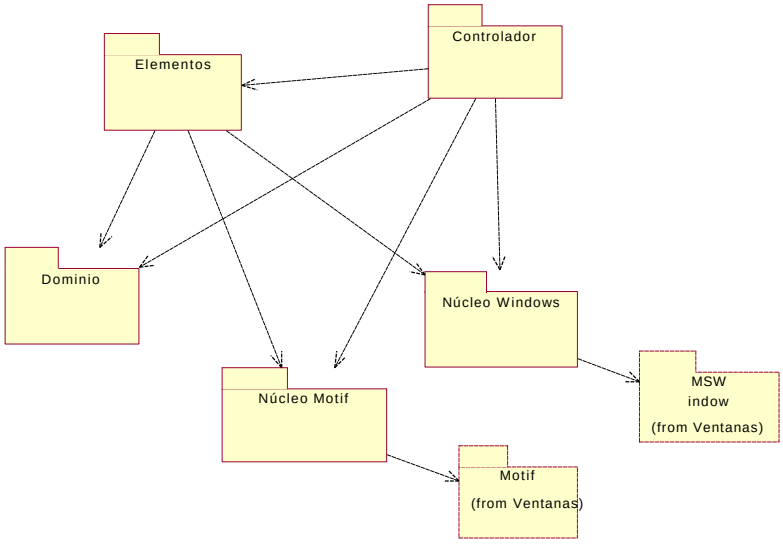


Figura 3.3 *Diagrama Editor*

3.3 Actividad 2

Estando en el navegador sobre el paquete de la Vista de Casos de Uso, con el botón derecho del ratón haga **new-package** y cree un paquete que se llame **Actividad 2**.

Con el botón derecho del ratón y estando en el navegador sobre el paquete recién creado haga **new-Use Case Diagram** y cree un diagrama que se llame **Actividad 2**.

Dibuje en el diagrama **Actividad 2** lo mostrado en la figura 3.3.

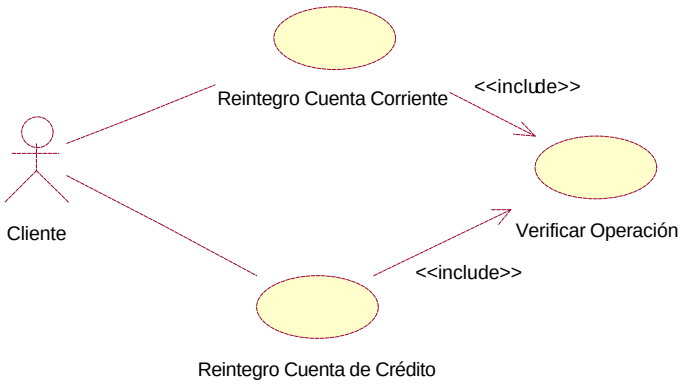


Figura 3.3: *Diagrama Actividad 2*

Observaciones:

Los estereotipos se introducen en la especificación del símbolo de generalización (hacer doble clic sobre el símbolo para abrir su especificación)

La opción Navigable establece la dirección en una asociación (puede habilitarse o deshabilitarse con el botón derecho sobre el símbolo)

3.4 Actividad 3

Estando en el navegador sobre el paquete de la Vista de Casos de Uso, con el botón derecho del ratón haga **new-package** y cree un paquete que se llame **Actividad 3**.

En el paquete recién haga **new-Use Case Diagram** y cree un diagrama que se llame **Actividad 3**. Dibuje en el diagrama **Actividad 3** lo mostrado en la figura 3.4.

Figura 3.4: *Diagrama Actividad 3*

Observación: Puede arrastrar el actor Cliente desde el paquete Actividad 2.

Con el botón derecho del ratón y estando en el navegador sobre el Caso de Uso **Reintegro** haga **new-Sequence Diagram** y cree un diagrama que se llame **Reintegro Saldo Insuficiente**.

Haga doble clic en el diagrama **Reintegro Saldo Insuficiente** y dibuje el diagrama mostrado en la Figura 3.5

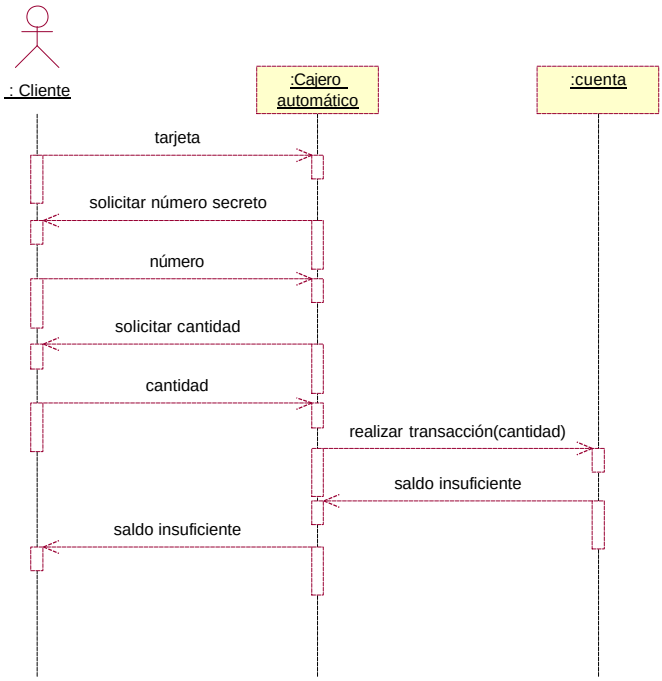


Figura 3.5: *Diagrama Reintegro Saldo Insuficiente*

Haga **Browse-Create Collaboration Diagram** para obtener automáticamente el Diagrama de Colaboración asociado.

3.5 Actividad 4

Crear el paquete **Actividad 4** en la Vista Lógica.

Dentro de este paquete crear las clases: **avión, motor, avión militar, avión comercial, vuelo, piloto, reserva, línea aérea, avión de carga, avión de pasajeros, vendedor de billetes.**

Cree dentro de la **Actividad 4** el Diagrama de Clases **Actividad 4**, mostrado de la Figura 3.6.

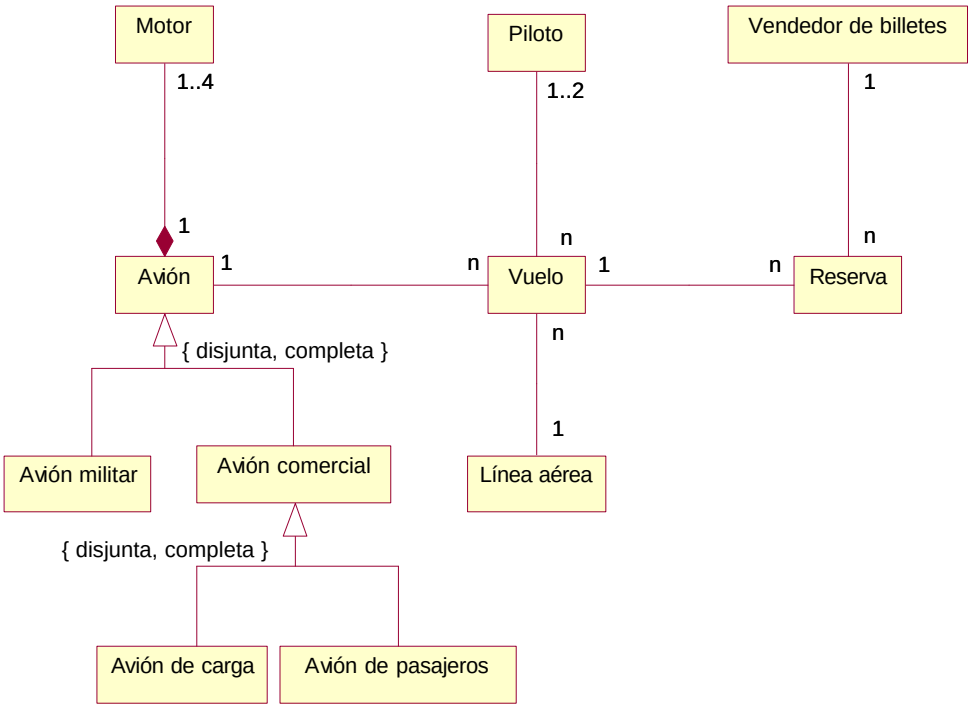


Figura 3.6: *Diagrama Actividad 4*

3.6 Actividad 5

En la Vista Lógica cree el paquete **Actividad 5**. Dentro de este paquete cree un Diagrama de Clases que se llame **Actividad 5**.

Incluya una única clase dentro de este diagrama que se llame Alumno y complete según lo mostrado en la Figura 3.7.

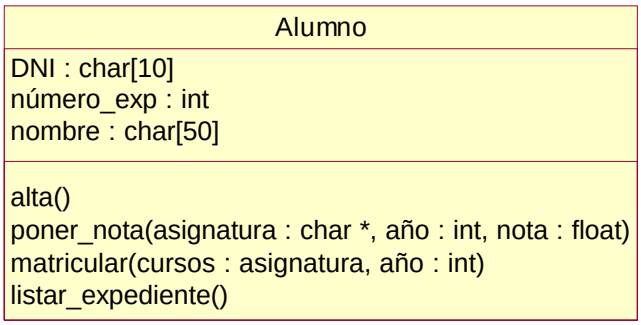


Figura 3.7: *Diagrama Actividad 5*

Observación: Pregunte al profesor si no consigue obtener la presentación mostrada en la Figura 3.7.

3.7 Actividad 6

En la Vista Lógica cree un paquete denominado **Actividad 6**.

Asociado al paquete **Actividad 6** cree el Diagrama de Clases **Actividad 6** e inserte las clases **Departamento** y **Profesor** y asócielas tal como se muestra en la Figura 3.8.

Modifique la visibilidad de los roles eligiendo entre **Público** (+): el rol es visible fuera del ámbito del paquete y puede referenciarse en otras partes del modelo; **Implementación** (sin símbolo asociado): visible sólo en el paquete en el que se define; **Protected** (#): accesible a la clase misma, a las subclases o *friends*; **Private** (-): accesible solo a la propia clase o *friends*.

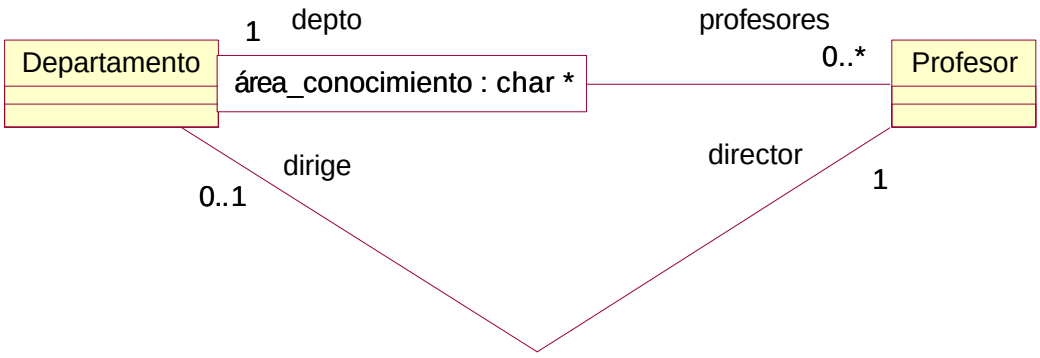


Figura 3.8: *Diagrama Actividad 6*

3.8 Actividad 7

Cree el paquete Actividad 7 y dentro de él introduzca el diagrama de clases **Actividad 7** con las clases **Empresa**, **Empleado** y **Cargo**. Defina en la clase **Cargo** los atributos **Nombre** y **Sueldo**.

Establezca la asociación entre **Empresa** y **Empledo**, mostrada en la figura 3.9.

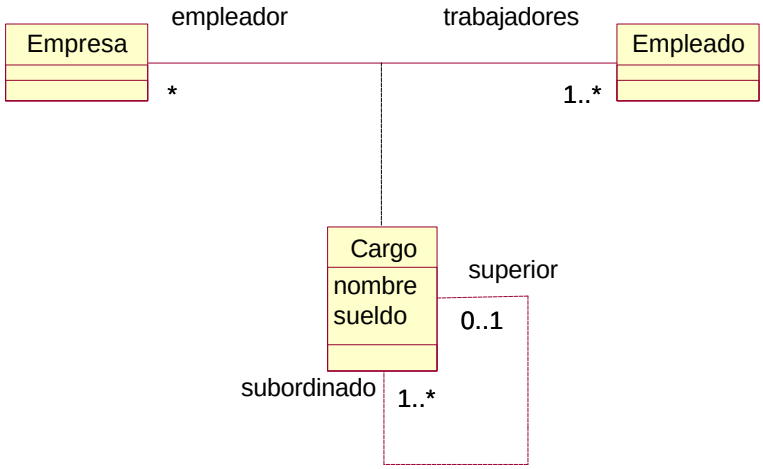


Figura 3.9: *Diagrama Actividad 7*

Observación: Use el símbolo de la barra de herramientas denominado "Link Attribute" para enlazar la clase **Cargo** con la asociación entre **Empresa** y **Empleado**.

3.9 Actividad 8

Cree el paquete **Actividad 8**.

Cree en el navegador las clases: **Trabajador**, **Directivo**, **Administrativo**, **Obrero**, **Vehículo**, **Vehículo impulsado por viento**, **Vehículo Terrestre**, **Vehículo impulsado por motor**, **Vehículo acuático**, **Camión**, **Velero**, **Cuenta**, **Cuenta rentable** y **Cuenta no rentable**.

Cree el Diagrama de Clases llamado **Actividad 8.1** según se muestra en la Figura 3.10.

Repita la operación para las Figuras 3.11 y 3.12.

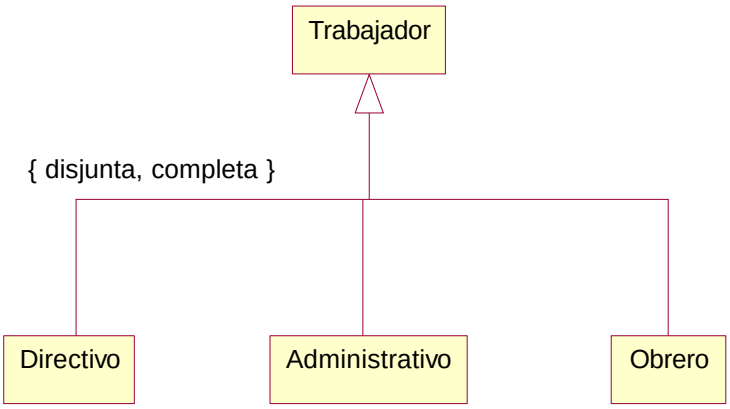


Figura 3.10: *Diagrama Actividad 8.1*

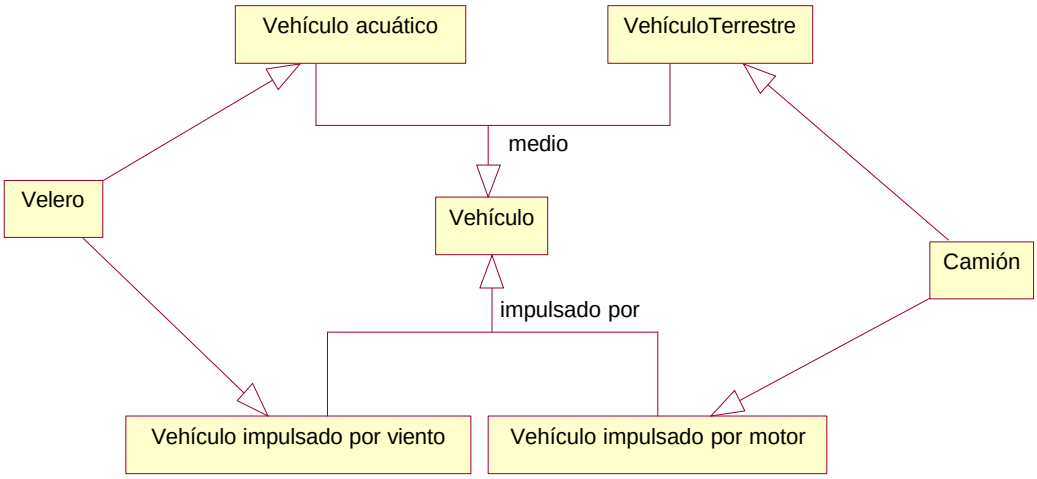


Figura 3.11: *Diagrama Actividad 8.2*

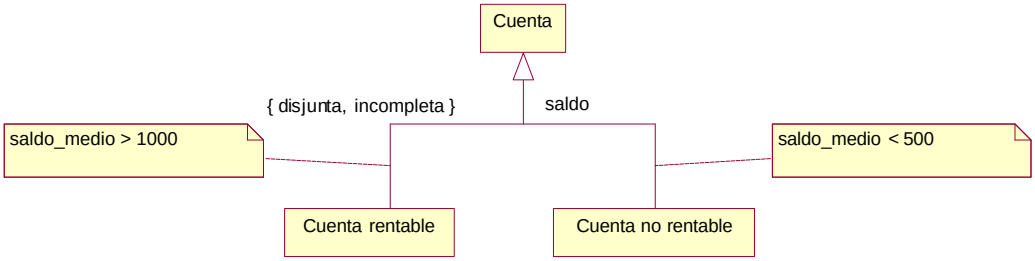


Figura 3.12: *Diagrama Actividad 8.3*

3.10 Actividad 9

Cree el paquete **Actividad 9**.

Cree en este paquete la clase **Socio** en un Diagrama de Clases que se llame **Actividad 9**. La Figura 3.13 da el detalle de la estructura de la clase.

Asocie a la clase anterior el Diagrama de Transición de Estados de la Figura 3.14. Para ello, desde el navegador seleccionando la clase en cuestión y con el botón derecho del ratón escoja la opción **Open State Diagram**.

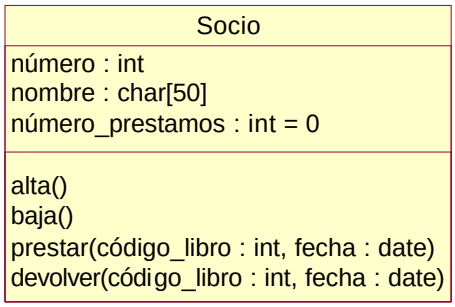


Figura 3.13: *Diagrama Actividad 9*

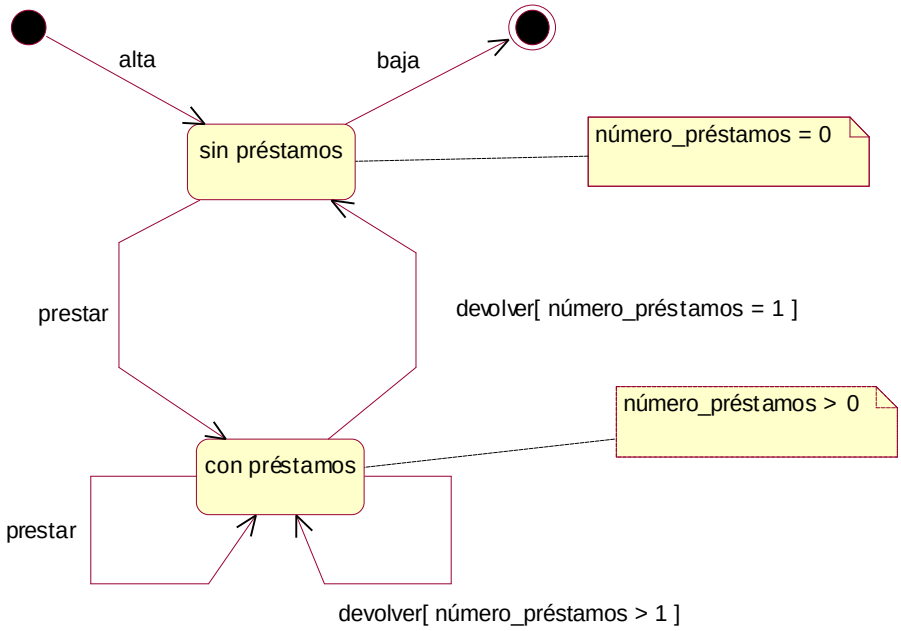


Figura 3.14: *Diagrama de Estados*

3.11 Actividad 10

Cree en la Vista de Componentes un paquete que se llame **Actividad 10** y dibuje el diagrama que se muestra en la Figura 3.15. Una relación de dependencia entre componentes viene dado porque un componente usa las facilidades de otro. Esto se reduce a dependencias de compilación entre componentes. Consulte en el Help los estereotipos para los componentes.

Dibuje el Diagrama de Despliegue de la Figura 3.16. Una **Connection** representa p.e. un cable RS232, comunicación vía satélite, etc. Un **Processor** representa hardware con capacidad de computación. Un **Device** incluye dispositivos hardware como terminales, modems, etc.

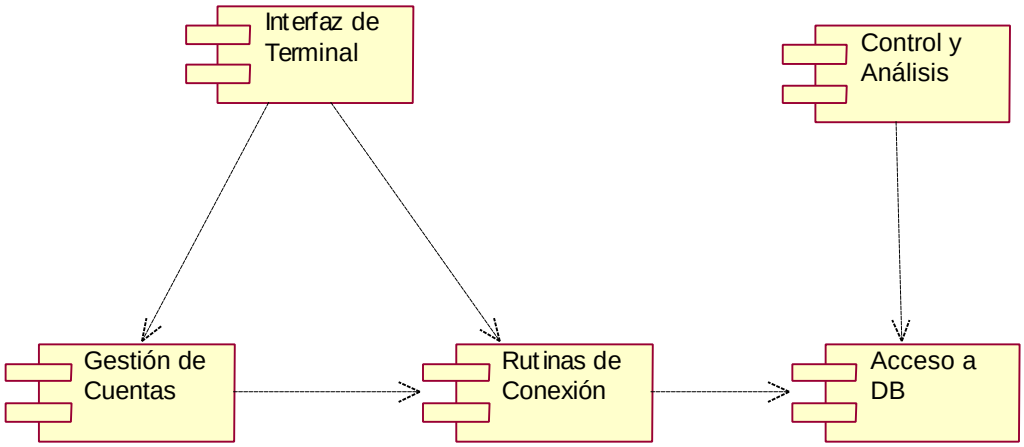


Figura 3.15: *Diagrama de Componentes*

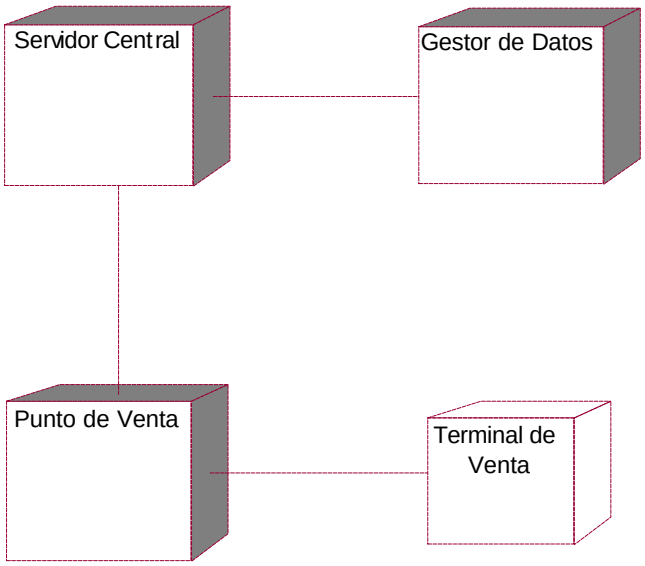


Figura 3.16: *Diagrama de Despliegue*

3.12 Actividad 11

Cree un nuevo modelo y renombre el diagrama **Main** de la Vista de Casos de Uso por **ACME**.

Haga doble click sobre el icono del diagrama **ACME** y dibujando, introduzca los subpaquetes **Publicidad**, **Ventas**, **Inventario** y **Contabilidad**. El resultado se muestra en la Figura 3.17

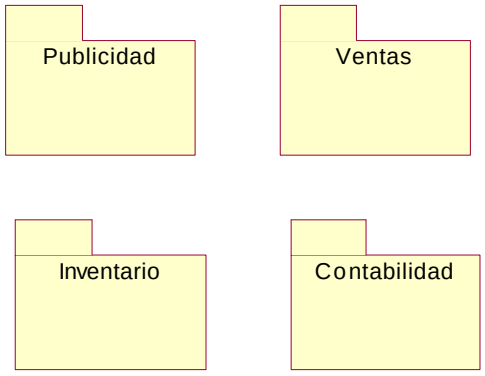


Figura 3.17: *Diagrama ACME*

Haga doble click sobre el paquete **Ventas** en el Diagrama **ACME** e introduzca el diagrama de casos de uso mostrado en la Figura 3.18.

Con el botón derecho sobre el diagrama llamado **Main** bajo el paquete **Ventas** renómbrelo por **Ventas**.

Asociado al paquete **Realizar Venta** crear un diagrama de casos de uso llamado **Realizar Venta**. Hacer doble click sobre el icono que representa el paquete **Realizar Venta** e introduzca el diagrama mostrado en la Figura 3.19

Renombre como **Realizar Venta** el diagrama **Main** bajo el paquete **Realizar Venta**. El resultado hasta este punto puede verse en la Figura 3.20.

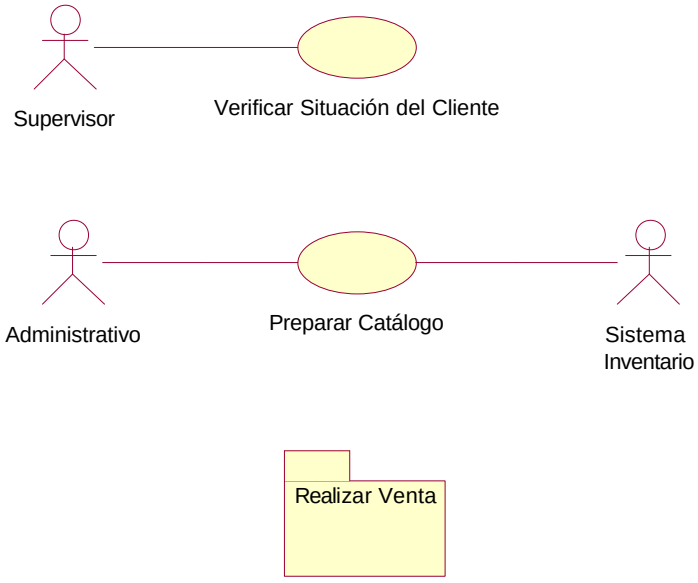


Figura 3.18: *Diagrama Ventas*

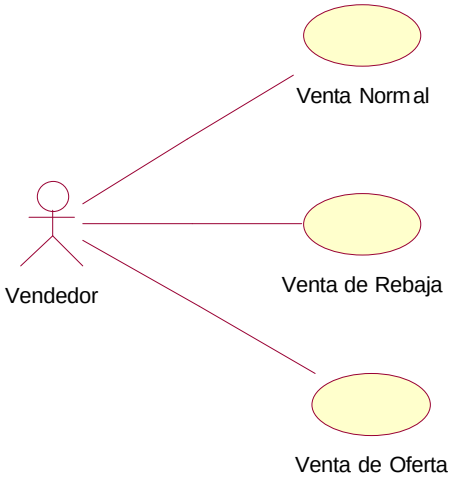


Figura 3.19: *Diagrama Realizar Venta*

En los D. de Casos de Uso no existe el concepto de “explosión” tal como se tiene en los DFDs (Diagramas de Flujo de Datos). La funcionalidad representada por un caso de uso es “atómica” (aunque en Rational Rose 98 a un caso de uso se le puede asociar un nuevo D. de Casos de Uso!!). En UML el concepto de paquete permite organizar de manera jerárquica un modelo, y en este caso, un paquete puede tener asociado un nuevo diagrama.

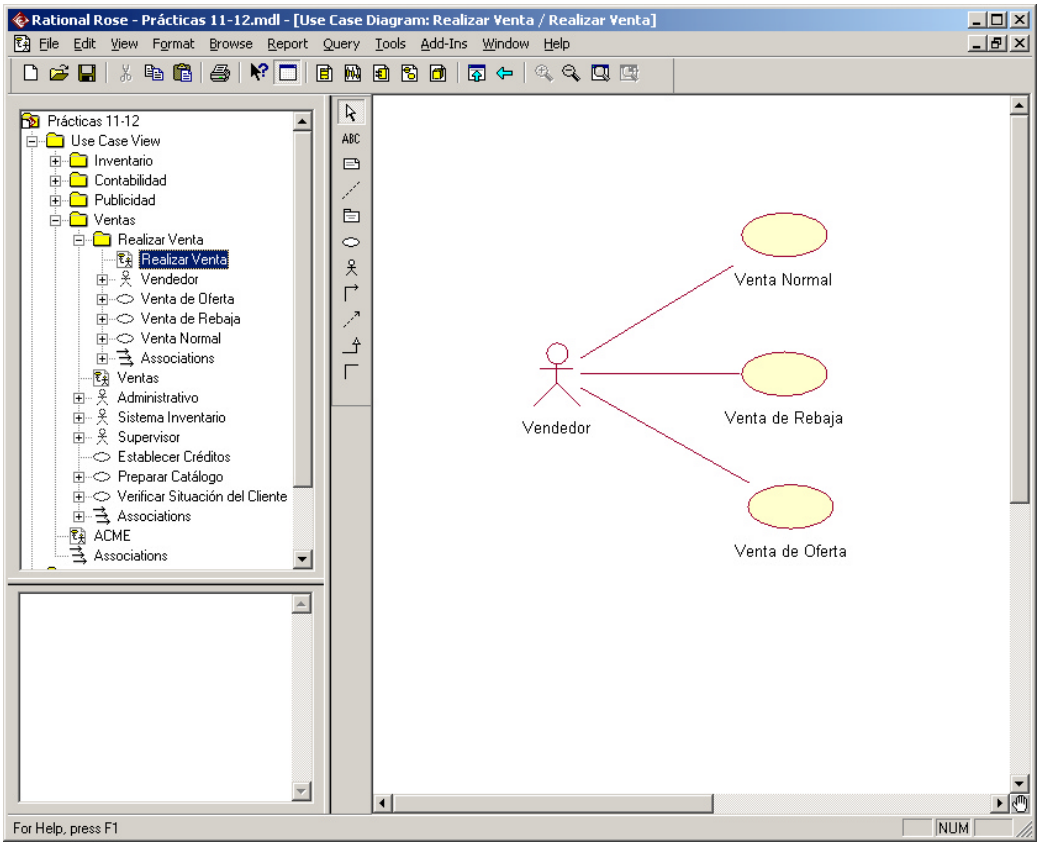


Figura 3.20: *Estado de la Práctica al terminar el paso f)*

Documente los casos de uso **Venta Normal**, **Venta Rebajas**, **Venta Ofertas** a partir de la información siguiente, presentada en tres estilos distintos ("secuencia de pasos", "condiciones pre-post de la aplicación del caso de uso" y, por último "descripción narrativa").

Venta Normal

Cree un fichero word con el siguiente contenido:

Caso de Uso Venta Normal

El cliente se identifica mostrando su tarjeta y el DNI

El vendedor revisa los datos del cliente

El vendedor introduce su código de vendedor e indica al sistema que se trata de una venta normal

El sistema muestra la pantalla para introducir los datos de la venta

El vendedor introduce los artículos mediante un lector de código de barras o directamente por teclado. Pueden ser varios artículos en una misma venta.

El vendedor solicita la emisión del recibo

El sistema imprime el recibo

Haga doble click sobre el caso de uso **Venta Normal** del diagrama y en la pestaña Files con el botón derecho realice Insert File, asociando el fichero word recién creado.

Venta en Oferta

Haciendo doble click en el caso de uso Venta en Oferta y dentro del cuadro denominado documentación, introducir:

Precondiciones

- Los artículos de la venta deben estar en oferta

- El pago debe hacerse en efectivo

- El artículo debe tener el suficiente stock para satisfacer la venta

Postcondiciones

- El stock del artículo se decrementa con la venta realizada

- Se registran todos sus datos en la base de datos

Venta en Rebajas

Seleccionando el caso de uso **Venta en Rebajas**, introducir en el cuadro de documentación (bajo el browser) el siguiente texto:

En el periodo de rebajas los precios tienen una disminución de precio tanto de forma individual como por grupos de artículos. Los descuentos se detallan en la correspondiente tabla de descuentos por grupo.

3.13 Actividad 12

Cree un nuevo modelo y renombre el diagrama **Main** de la Vista de Casos de Uso por **Video Club**.

Introduzca en el Diagrama **Video Club** el modelo de la figura 3.21.

Figura 3.21: *Diagrama Video Club*

Cree un Diagrama de Secuencia asociado al Caso de Uso **Prestar Video** y denomínelo **Prestar con Éxito**. Arrastre desde el navegador el actor **Encargado** y complete el Diagrama de Secuencia según lo mostrado en la Figura 3.22. Los objetos utilizados en este diagrama son anónimos, es decir, sólo se indica la clase a la cual pertenecen, pero no se les asigna un nombre específico.

Deshabilite la opción **Focus of Control** en **Tools-Options-Diagrams** y observe el efecto.

Cree el Diagrama de Colaboración asociado al Diagrama de Secuencia dibujado mediante **Browse-Create Collaboration Diagram**. La Figura 3.23 muestra el diagrama de colaboración que se debe obtener.

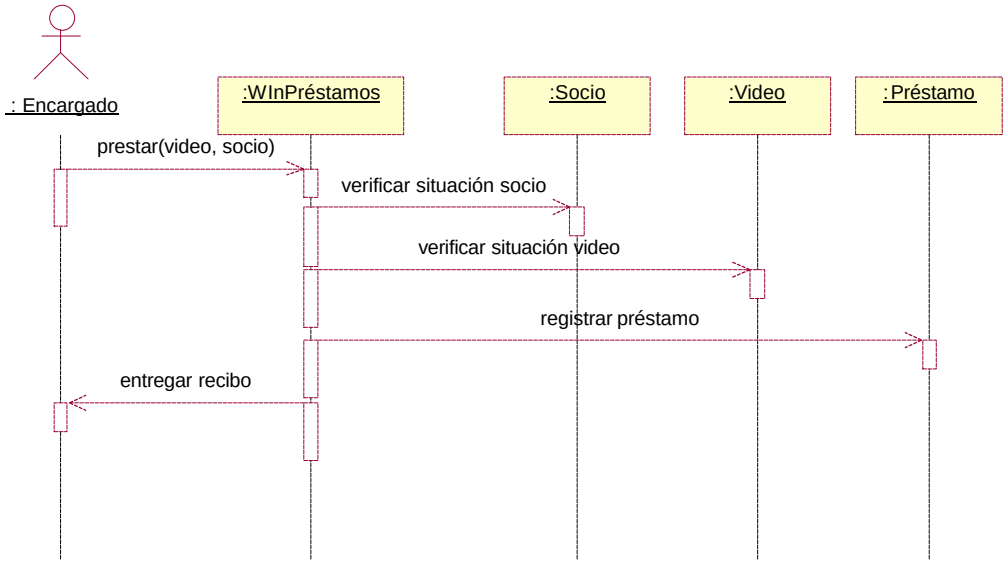


Figura 3.22: *Diagrama Prestar con Éxito*

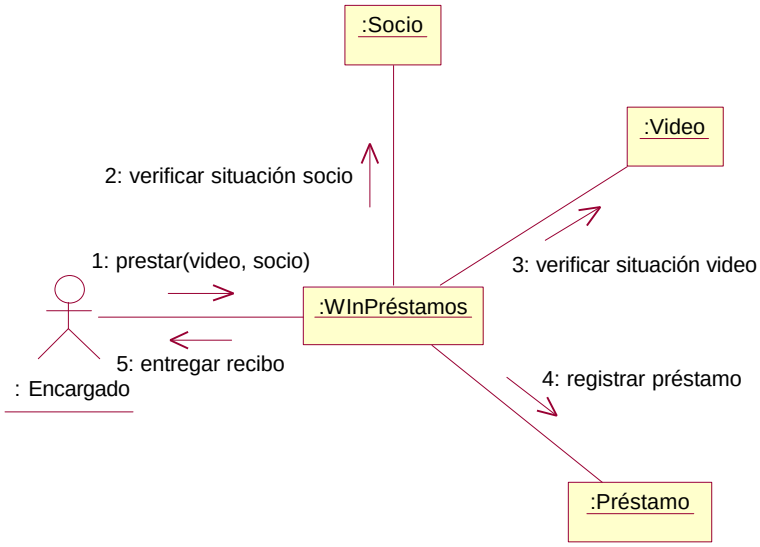


Figura 3.23: *Diagrama Obtenido a partir del Diagrama Prestar con Éxito*

Tema 4. Actividades para desarrollo de la práctica

4.1 Introducción

La siguiente es una lista de actividades, que puede servir como pauta general de desarrollo usando UML. Hay que insistir en el hecho que dichas actividades deberían retroalimentarse entre ellas en un proceso de refinamiento sucesivo.

Comenzar con el análisis del negocio y Casos de Uso

Capturar cómo funciona actualmente el negocio y cómo se desea que funcione. Usar la técnica Casos de Uso para construir escenarios[1] que modelen los procesos del sistema. Construir un diagrama de Casos de Uso para cada escenario en el sistema.

Migrar desde los diagramas de Casos de Uso a diagramas de Secuencia y diagramas de Colaboración

Realizar un diagrama de secuencia o colaboración por cada Caso de Uso, capturando los objetos y los mensajes entre ellos. La secuencia de pasos del Caso de Uso se traduce en una secuencia de interacciones entre objetos en el diagrama de secuencia. El diagrama de Colaboración muestra las mismas interacciones, pero sin indicar su secuencia. Es una visión alternativa para el diagrama de Secuencia.

Construir el diagrama de Clases

Especificar la estructura de las clases y sus relaciones de herencia. Los objetos modelados en los diagramas de Secuencia y Colaboración son utilizados para modelar las clases en el diagrama de Clases.

[1] Un escenario es el planteamiento de una situación de funcionamiento del sistema, incluye la determinación de los actores externos y de la parte del sistema involucrada. Un escenario puede ser descrito por un Caso de Uso.

Construir diagramas de Estado

Modelar el comportamiento de un objeto particular o clase de objetos. Realizar un diagrama de Estado para cada clase que tenga un comportamiento significativo.

Modelar componentes de software

Utilizar el diagrama de Componentes para modelar la estructura del software, incluyendo dependencias entre componentes de software, componentes en código binario y componentes ejecutables.

Modelar la distribución e implementación

Utilizar el diagrama de Despliegue para modelar la configuración en tiempo de ejecución de los componentes del sistema, modelando nodos y comunicación entre ellos.

Tema 5. Casos de Uso. Ejercicios Resueltos

Ejercicio 1. Gestión de fincas e inmuebles

Enunciado

Se desea desarrollar una aplicación de gestión de fincas e inmuebles. La aplicación deberá cubrir todos los aspectos relacionados con dicho tema, teniendo en cuenta la siguiente dinámica de funcionamiento:

Una empresa gestiona un conjunto de inmuebles, que administra en calidad de propietaria. Cada inmueble puede ser bien un local (local comercial, oficinas, ...), un piso o bien un edificio que a su vez tiene pisos y locales. Como el número de inmuebles que la empresa gestiona no es un número fijo, la empresa propietaria exige que la aplicación permita tanto introducir nuevos inmuebles, con sus datos correspondientes (dirección, número, código postal, ...), así como darlos de baja, modificarlos y consultarlos. Asimismo, que una empresa administre un edificio determinado no implica que gestione todos sus pisos y locales, por lo que la aplicación también deberá permitir introducir nuevos pisos o locales con sus datos correspondientes (planta, letra,...), darlos de baja, modificarlos y hacer consultas sobre ellos.

Cualquier persona que tenga una nómina, un aval bancario, un contrato de trabajo o venga avalado por otra persona puede alquilar el edificio completo o alguno de los pisos o locales que no estén ya alquilados, y posteriormente desalquilarlo. Por ello deberán poderse dar de alta, si son nuevos inquilinos, con sus datos correspondientes (nombre, DNI, edad, sexo, fotografía, ...), poder modificarlos, darlos de baja, consultar, etc. (para la realización de cualquiera de estas operaciones es necesaria la identificación por parte del inquilino). Por otra parte, cada mes el secretario de la empresa pedirá la generación de un recibo para cada uno de los pisos y de los locales, el cual lleva asociado un número de recibo que es único para cada piso y

para cada local y que no variará a lo largo del tiempo, indicando el piso o local a que pertenece, la fecha de emisión, la renta, el agua, la luz, la actualización del IPC anual, portería, IVA, etc. Y otros conceptos, teniendo en cuenta que unos serán opcionales (sólo para algunos recibos) y otros obligatorios (para todos los recibos). Además, para cada recibo se desea saber si está o no cobrado.

Con vistas a facilitar la emisión de recibos cada mes, la aplicación deberá permitir la generación de recibos idénticos a los del mes anterior, a excepción de la fecha. Además deberán existir utilidades para inicializar los conceptos que se desee de los recibos a una determinada cantidad y también debe ser posible modificar recibos emitidos en meses anteriores al actual. La aplicación también deberá presentar los recibos en formato impreso, pero teniendo en cuenta que en un recibo nunca aparecerán aquellos conceptos cuyo importe sea igual a cero.

De igual forma, el secretario debe poder gestionar los movimientos bancarios que se producen asociados a cada edificio, piso o local. Un movimiento bancario siempre estará asociado a un banco y a una cuenta determinada de ese banco. En esa cuenta existirá un saldo, acreedor o deudor, que aumentará o disminuirá con cada movimiento. Para cada movimiento se desea saber también la fecha en que se ha realizado. Un movimiento bancario puede ser de dos tipos: un gasto o un ingreso.

Si el movimiento bancario es un gasto, entonces estará asociado a un inmueble determinado, y se indicará el tipo de gasto al que pertenece entre los que se tienen estipulados. Ejemplos de gastos son el coste de la reparación de un ascensor del inmueble que pertenece a gastos de reparación, el sueldo de la señora de la limpieza, etc. Sí el movimiento bancario es un ingreso entonces estará asociado a un piso de un inmueble determinado o a un local y también se indicará el tipo de ingreso al que pertenece, como en el caso de los gastos. Ejemplos de ingresos son precisamente los recibos que se cobran cada mes a los inquilinos.

Basándose en los gastos e ingresos que se deducen de los movimientos bancarios, la aplicación deberá ser capaz de ocuparse de la gestión económica generando los informes que facilitan la realización de la declaración de la renta.

Por último, la aplicación deberá ser capaz de proporcionar el acceso, de forma estructurada, a toda la información almacenada en el sistema, generando para ello los listados necesarios que requiere el secretario.

Ejemplos de listado son: el listado de todo los inquilinos ordenado por fechas, el listado de inquilinos que han pagado o no en un determinado intervalo de tiempo, el listado de todos los inmuebles, el listado de todos los pisos y locales de cada edificio, el listado de todos los recibos pendientes de cobro en un determinado intervalo de tiempo, etc.

Solución:

A continuación se muestra el diagrama de casos de uso en el que se representan al actor propietario y las tareas requeridas por el sistema de gestión de fincas e inmuebles (ver Figura 5.1).

El propietario será el responsable de la gestión del edificio, local o de los pisos mediante el alta, las modificaciones, consulta y la baja de los mismos.

En este diagrama de casos de uso asociado con el propietario, los casos de usos con los que se comunica el actor son:

? *Gestión de edificios.*

? *Gestión de locales.*

? *Gestión de pisos.*

Cada uno de los casos de uso anteriores refleja las actividades comunes que se deben realizar en el alta, baja, modificación y consulta. Ya que en el enunciado se hace referencia a estas cuatro funcionalidades que se deben permitir en el sistema, se ha reflejado tal situación introduciendo un caso de uso específico si se hace referencia al edificio, al local o al piso. Se ha hecho este desglose y diferenciación dependiendo de si es un edificio, local o piso, ya que las operaciones que conllevan cada uno es distinta aunque podamos nombrarlas de la misma forma. Los datos y actividades que se manejan son diferentes.

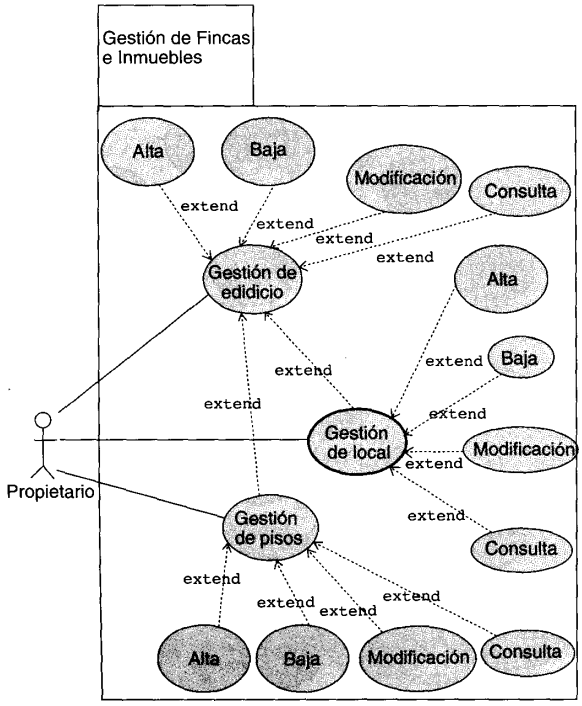


Figura 5.1:.*Casos de uso relacionados con el actor propietario.*

La relación empleada para organizar los casos de uso es la de un *extend,* ya que se intenta identificar que cualquiera de estas funcionalidades se pueden o no realizar tanto individual corno conjuntamente. Además, hemos relacionado mediante un *extend* el caso de uso de *Gestión de locales y de pisos* con el caso de *uso Gestión de edificio.* Con esto reflejamos que la gestión de edificios puede conllevar la gestión de locales, de pisos o de ambos.

En el siguiente diagrama (ver Figura 5.2) se muestran los casos de uso relacionados con el actor inquilino. El inquilino va a ser aquella persona que tiene algún tipo de aval, de los expuestos en el enunciado, y, por tanto, puede realizar algunas de las siguientes operaciones en el sistema:

? Alquilar.

? Desalquilar.

? Darse de baja.

? Modificar sus datos.

? Consultarlos.

Para cada una de estas operaciones hay un caso de uso en el diagrama reflejando la situación anterior. Además, ya que se nos dice que para la realización de cualquiera de las operaciones es necesaria su identificación, se ha reflejado un caso de uso nombrado *Identificación* que se relaciona con los anteriores mediante la relación de *include.* Con la relación de *include* hacemos especial énfasis en esta situación.

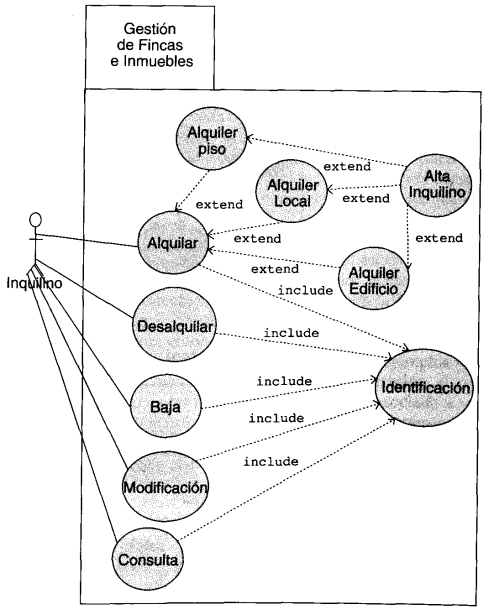


Figura 5.2: *Casos de uso del actor 'Inquilino".*

Tras volver a examinar con más detalle la descripción proporcionada se observa que cuando se produce el alquiler éste puede ser el de un piso *(Alquiler Piso)* un local *(Alquiler Local)* y de edificio *(Alquiler de* Edificio). Por ello se generan tres nuevos casos de uso que implican una relación de *extend* con el caso de uso de Alquilar.

Como hemos observado que la primera vez que se produce una operación de alquiler se debe permitir el alta de los datos del inquilino, se ha creado el caso de uso *Alta Inquilino* como una extensión de Alquiler Piso, Alquiler Local y Alquiler Edificio.

Finalmente, el último diagrama de caso de uso que se muestra (Figura 5.3) es aquel en el que se encuentra involucrado el actor *secretario.* Tras una visión general de las características del sistema, observamos que las tareas del secretario son:

Obtención de los distintos tipos de recibos. Obtener los informes económicos. Generación de los listados.

Como vemos, en aras de reflejar de una forma más meticulosa las funcionalidades que debe contemplar el sistema, todos los casos de uso genéricos, con los cuales está relacionado, se desglosan en otros casos de uso. Para ello se ha utilizado la relación de extensión en algunos casos de uso.

Así pues, el caso de uso de "Generar recibos" está relacionado mediante un *extend* con los casos de uso:

- ✍ Recibos idénticos mes anterior.
- ✍ Inicializar conceptos.
- ✍ Modificar los del mes anterior.

Este desglose se ha realizado para reflejar lo que el enunciado muestra con detalle y así poder tener una comprensión mayor de lo que el sistema debe de hacer.

Por otra parte, la *Gestión de movimientos bancarios* se extiende en los casos de uso de *Ingresos y Gastos de inmuebles,* mientras que el de *ingresos* se extiende en *ingresos de pisos* e *ingresos de local.* De esta forma reflejamos el hecho de que los ingresos pueden ser de pisos, de locales o ambos, pero sólo por esos conceptos.

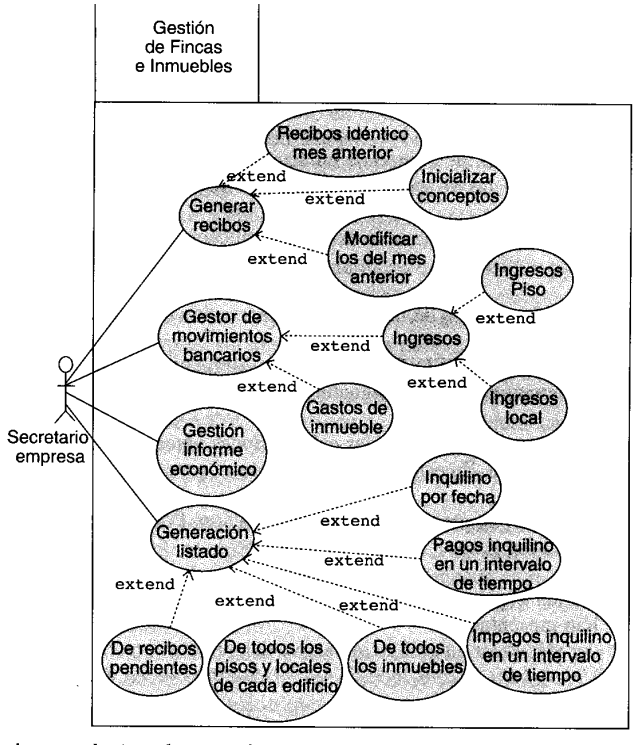


Figura 5.3:*Casos de uso relacionados con el actor "Secretario empresa".*

Finalmente, el caso de uso de *Generación de listados* se extiende en distintos casos de uso dependiendo del tipo de listado que se ha comentado en el enunciado. Con esto se indica claramente cuáles son específicamente las operaciones que se deben poder realizar, obteniendo, por tanto, una mayor comprensión de los requisitos que debe tener el sistema. La extensión refleja esos comportamientos opcionales que puede haber en el sistema y que no tienen por qué ser exclusivos, en el sentido de que si se realiza uno se pueden realizar los otros, cuando se *generan listados.*

Los casos de uso que tenemos son: *Inquilino por fecha, Pagos inquilino en un intervalo de tiempo, Impagos inquilino en un intervalo de tiempo, De todos los inmuebles, De todos los pisos y locales de cada edificio, De recibos pendientes.*

Ejercicio 2. Gestión calificaciones *Enunciado:*

Enunciado

Se desea desarrollar una aplicación de gestión de las calificaciones de los alumnos para satisfacer las numerosas quejas de los profesores, por el uso del lápiz y papel. La aplicación deberá cubrir únicamente aquellos aspectos relacionados con dicho tema, y que se describen a continuación:

El profesor recibe las actas en blanco de las asignaturas de las que es responsable, en formato electrónico. El acta contiene los siguientes datos de la asignatura (titulación, campus, curso académico, denominación de la asignatura, convocatoria y grupo) y la

lista de alumnos matriculados (niu, nif, nombre y apellidos). Alguna de las acciones que puede hacer el profesor son:

? Completar un acta con las notas de los alumnos.

? Añadir o borrar un alumno de un acta.

? Integrar las actas de varios grupos de una misma asignatura en una sola acta.

Otras de las opciones que se le exige a la aplicación, para satisfacer completamente las necesidades del profesor, son las siguientes:

? Permitir la consulta de la siguiente información de cualquier alumno seleccionado:

- DNI, N.° EXPEDIENTE, Lista de asignaturas en las que está matriculado el alumno (Código asignatura-Nombre asignatura).

? Obtener una estadística de las calificaciones obtenidas por los alumnos en un determinado grupo de una asignatura. En esta estadística se tendrá para cada posible calificación:

- Número de personas con esa calificación, Porcentaje sobre los presentados, Porcentaje sobre el total del grupo.

? Consultar el porcentaje de personas sobre el total del grupo que se han presentado y el de los que no se han presentado.

? Poder visualizar un gráfico indicativo del número de personas que han obtenido una calificación entre 0-0.99, 1-1.99, 2-2.99, 3-3.99, 4-4.99, 5-5.99, 6-6.99, 8-8.99, 9-10; indicándose la nota media obtenida por la clase.

? Disponer de una calculadora que permita realizar las operaciones de suma, resta, multiplicación, división. Esta calculadora se activará cuando se vayan a introducir las notas a algún alumno de forma que una vez realizada la operación aritmética, pulsando un botón se vuelque el resultado en la casilla donde se están introduciendo las calificaciones, redondeándose a dos cifras decimales.

? Permitir la importación y exportación de la lista de alumnos con sus calificaciones a un formato compatible con MS Excel.

? Imprimir las actas y la lista provisional de calificaciones.

Finalmente, como una ampliación extra, a la cual sólo podrá acceder quien se identifique inicialmente como administrador de la aplicación, se deben permitir:

? Gestión ABMC (Altas/Bajas/Modificación y Consulta) de los datos de un alumno y su matriculación en una asignatura y a un grupo.

? Gestión de Asignaturas, teniendo en cuenta que una asignatura sólo se puede dar en un único curso (primero, segundo, tercero...) y que cada curso está

formado ponlos datos sobre el número máximo de alumnos, número mínimo de créditos troncales y número mínimo de créditos optativos. Algunos de los datos que vamos a poder consultar de una asignatura son el nombre, número de créditos y cuatrimestre en el que se imparte.

? Gestión de Titulaciones, teniendo en cuenta que una titulación sólo se da en un campus determinado y los datos que podemos consultar son el nombre, el número de créditos o carga lectiva global, si es de 1.° o 2.' ciclo, ...

? Gestión de grupos, en los que podemos consultar el número máximo de alumnos permitidos, si es un grupo de mañana, de tarde o de noche, y cuál es el código empleado para identificar el grupo.

? Consultar aquellos alumnos que no se pueden matricular y el motivo de ello.

? Consultar el historial académico de un alumno.

Solución

A continuación se muestra el diagrama de casos de uso en el que se representan al actor profesor junto con las tareas que requiere del sistema de gestión de calificaciones (ver Figura 5.4). Así tenemos que:

El profesor será aquel que puede realizar una serie de operaciones relacionadas con el listado de alumnos que tiene matriculados en sus asignaturas, tales como introducir las notas de alumnos, consultar el historial de alguno de sus alumnos, introducir o eliminar algún alumno en el listado y tareas de estadística y de importación y exportación.

Se ha intentado reflejar toda la funcionalidad del sistema asociada al actor profesor para poder mostrar qué es lo que se espera que haga el sistema de forma completa.

Así pues, se tiene el caso de uso de Poner notas, el cual se extiende en el caso de uso de Operaciones Calculadora. Con ello se refleja que el profesor al introducir las notas puede en algún momento hacer uso de las operaciones que aporta una calculadora. Y ya que actualmente una calculadora ofrece una gran variedad de operaciones se han detallado mediante la relación de *extend* las operaciones que el profesor podría utilizar, como son: Sumar, Restar, Multiplicar o Dividir. Finalmente, para completar cuál es la funcionalidad completa que se espera que permita el caso de uso de Operaciones Calculadora se identifica el caso de uso de Volcar Resultado mediante la relación de *include*, ya que es algo que debe poder realizar siempre que se haga alguna operación con la calculadora.

Por otra parte, el caso de uso de Gestión de alumno se ha relacionado con el caso de uso de Añadir y Borrar mediante un *extend* para identificar explícitamente cuáles son las acciones que se espera que permita el sistema y

que o bien se puede realizar de forma individual o no, cuando el profesor utiliza el sistema.

Otras de las funcionalidades que constituyen un caso de uso cada una son: Integrar grupos, Información alumno, Estadística, Gráfico, Importar y Exportar.

De la misma forma que anteriormente hemos comentado, se ha realizado el proceso de identificación explícita de las operaciones que se pueden realizar cuando el profesor quiere Imprimir. Se ha expresado mediante el *extend* las formas de impresión que puede tener el profesor reflejando que cuando se imprime puede imprimir sólo las actas (Imprimir actas), sólo las listas (Imprimir Listas provisional) o ambas.

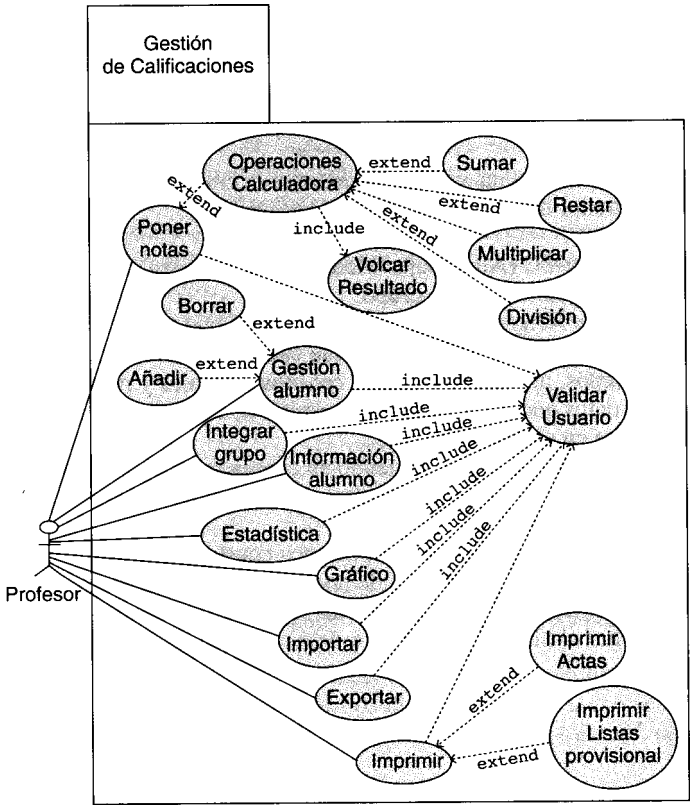


Figura 5.4: *Casos de uso relacionados con el actor "Profesor".*

Finalmente se muestra que todos los casos de uso con los que se relaciona de forma directa el actor se relacionan con el caso de uso de *Validar Usuario* para mostrar que es necesaria la identificación del profesor en el sistema para poder realizar cualquier operación comentada anteriormente.

En el siguiente diagrama se muestran todos los casos de uso relacionados con el actor administrador del sistema (ver Figura 5.5).

El administrador será el responsable del mantenimiento de los datos que hay en el sistema respecto a las asignaturas y a los alumnos matriculados.

Como podemos observar, se ha intentado expresar toda la funcionalidad del sistema y por ello se han desglosado al máximo todos los casos de uso hasta que cada uno refleja una funcionalidad del sistema.

En la siguiente figura se muestran cuáles son de forma explícita las funcionalidades que conlleva la gestión de los alumnos (caso de uso *Gestión ABMC Alumnos).* Así pues, se ha expresado mediante la relación de *extend* las distintas posibilidades que ofrece la gestión de alumnos, mostrándose además que ninguna es excluyente y que se pueden realizar algunas de las operaciones o todas cuando el administrador gestiona a los alumnos (casos de uso *Alta, Baja, Modificación, Consulta Historial Académico).*

El resto de funcionalidades que el administrador espera del sistema son las siguientes:

? *Matriculación,* que identifica a la capacidad del sistema para realizar la matriculación de un alumno en las asignaturas, titulaciones y grupos existentes.

? *Gestión de Asignaturas,* que identifica la posibilidad que tiene el administrador para introducir, borrar, modificar y consultar las asignaturas. En este caso no se ha reflejado de forma explícita, ya que en el enunciado no aparece detallado cuál es el alcance de la gestión de asignaturas.

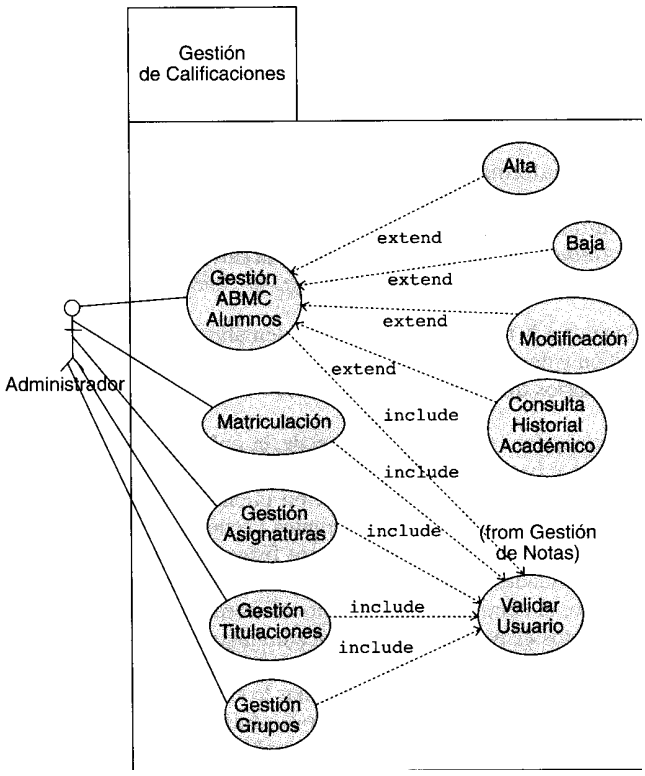


Figura 5.5: *Casos de uso relacionados con el actor "Administrador".*

? • *Gestión de Titulaciones y Gestión de Grupos* reflejan la posibilidad para que el *administrador* introduzca en el sistema los datos de titulaciones y

Grupos. Como ocurría anteriormente, al no detallarse en el enunciado del problema cuál es el alcance real de estas operaciones sólo se reflejan estos casos de uso sin el detalle mostrado para la *Gestión ABMC Alumnos.*

Resaltar el hecho de que el caso de uso de Validar Usuario esté relacionado, mediante la asociación de *include,* con todos los casos de uso con los que se relaciona directamente el actor *administrador.* De esta forma indicamos que cualquier administrador que tenga que realizar una tarea o función debe identificarse en el sistema. El caso de uso que aparece en el gráfico siguiente es el mismo que se identificó anteriormente.

Ejercicio 3. Puntos de información universitaria

Enunciado

La Universidad Carlos III de Madrid en su constante innovación pretende instalar un conjunto de Puntos de Información Universitaria (PIU) a través de los cuales se pueda facilitar información a la comunidad universitaria.

Las funcionalidades consideradas para instalar en cada PIU son:

? Información General: actividades culturales y extra-académicas de la Universidad y de las diferentes Escuelas y Facultades.

? Información Administrativa: plazos de matriculación, fechas de exámenes, normativas y avisos.

? Información Privada: esta información se diferenciará según el tipo de usuario final que se identifique en el PIU.

PAS: información relativa a su cuerpo e información económico-contractual.

Profesores: información relativa a su cuerpo, información de asignación horaria de clases e información económico-contractual.

Alumnos: información referente a la carrera que están cursando y su currículum, así como el estado de su matriculación.

Como ayuda a la resolución de esta problemática, la Universidad Carlos 111 ha pedido a su departamento de investigación y desarrollo (I+D) la elaboración de un sistema informático que pueda ser utilizado por cuatro tipos de usuarios diferentes:

? *Administrador:* es el responsable de la colocación y carga inicial de los PIU's en las diferentes Escuelas y Facultades que componen la Universidad, es decir, se encarga de decidir, las situaciones físicas más propicias y de activación inicial de los contenidos (funcionalidades a proporcionar) de cada uno de los PIU's en las diferentes Escuelas y Facultades que componen la Universidad, es decir, se encarga

de decidir las situaciones físicas más propicias y de activación inicial de los contenidos (funcionalidades a proporcionar) de cada uno de los PIU's.

Por tanto, el administrador tan sólo utilizará este sistema informático para notificar la instalación de los distintos dispositivos. Habrá un administrador de dispositivos por cada turno de mañana y de tarde para solucionar todas las peticiones realizadas por los responsables de cada centro.

? *Gestor:* es el encargado de determinar la situación (funcionamiento/desconexión) de cada uno de los PIU's distribuidos previamente por el administrador del sistema.

Asimismo, este usuario será el responsable de determinar qué acciones se desencadenarán como consecuencia de la aparición de un mal funcionamiento del PIU's, como puede ser:

-Registro en una salida de "Log". - Envío de un equipo técnico.

-Reporte del error al CAT (Centro de Atención Técnico).

-Reinicialización del PIU.

-Emisión de una solicitud de desconexión del PIU al administrador.

Como la principal misión de los gestores de los PIU's es la regulación y mantenimiento de los mismos, tan sólo utilizarán el sistema informático de forma esporádica, para retocar los parámetros de funcionamiento del sistema cuando se detectan anomalías a tener en cuenta. Habrá un gestor de dispositivos en el turno de mañana y en el de tarde.

? *Operador:* es el usuario responsable de gestionar el funcionamiento de cada uno de los PIU's existentes en cada una de las Escuelas y Facultades. Su actividad consistirá en el control de red, es decir, se encarga de verificar el funcionamiento global de la red de PIU's existente. Pudiendo realizar operaciones de control, gestión y estadísticas sobre la misma. Además, se encarga de reportar los errores observados al Gestor que esté de guardia en cada momento.

Los operadores estarán utilizando continuamente el sistema de seguimiento de los PIU's, tan sólo lo dejarán de utilizar en los periodos de descanso acordados. La Universidad utilizará a tres operadores en activo para cada uno de los turnos de servicio (mañana, tarde y noche).

Por último, los operadores también deberán realizar las acciones indicadas por el gestor del sistema en caso de que éste no esté localizable.

? *Usuarios Finales:* este grupo está compuesto por el PAS, el Profesorado y el Alumnado. Su conexión al sistema vendrá siempre asociada a una solicitud/servicio de información.

Cada vez que un usuario intente conectarse al sistema deberá introducir sus datos identificativos, así como la introducción de una contraseña y del tipo de usuario (en

caso de que sea necesario). Las actividades recogidas por el sistema sólo estarán accesibles para el tipo de usuario responsable de su realización, de tal manera, que la instalación de PIU's no estará accesible a un gestor o a un operador, del mismo modo la gestión de red no podrá ser realizada por un administrador o por un gestor.

Instalación de los PIU's

Para instalar un PIU dentro de una Facultad o Escuela será necesario, en primer lugar, seleccionar la Escuela/Facultad, de tal modo que sólo puede haber un único dispositivo de un tipo determinado en una misma Escuela/Facultad. A continuación se indicará las funcionalidades que soportará dicho PIU. Será posible que el administrador de los PIU's cambie la colocación de los mismos, así como el resto de características propias del PIU.

Control de funcionamiento

Periódicamente, el gestor de los PIU's podrá observar el estado de funcionamiento de cada uno de los PIU's así como ajustar las acciones a realizar qué se desencadenará como consecuencia de la aparición de un mal funcionamiento del PIU's.

Gestión de red

Se podrán realizar operación de control, gestión y estadística sobre la red instalada observando la aparición de errores, que deberán ser reportados al gestor de guardia.

Obtención de información

Los Usuarios Finales realizarán peticiones al sistema guiados a través de la interfaz gráfica del sistema, su única interrelación con el sistema, consiste en la emisión de dichas peticiones para que sean procesadas y servidas por el sistema.

Solución

A continuación se muestra el diagrama de casos de uso en el que se representan todos los actores y las tareas requeridas por el sistema de gestión de PIU's (ver Figura 5.6). Identificamos inicialmente a los actores que van a interactuar de alguna forma con el sistema, obteniendo la siguiente lista:

El *Administrador.*

El *Gestor.*

El *Operador o vigilante.*

El *Usuario final.*

Una vez que hemos identificado a los distintos usuarios registramos las operaciones que cada uno de ellos debe de poder realizar en el sistema. Así pues, indicamos las funcionalidades del sistema desde el punto de vista del usuario del sistema.

Así tenemos que:

?•El administrador será aquel que realice las tareas de *instalación* de los PIU's.

?•El gestor será el responsable de controlar el buen *funcionamiento* de los diferentes PIU's existentes en el sistema.

?El vigilante será el responsable de la *gestión de la red* en la que se encuentran los diferentes PIU's existentes en el sistema.

?El usuario final estará destinado a la realización de las *consultas* necesarias para la extracción de la información contenida en los PIU's.

Cada una de estas funcionalidades se representan como un caso de uso relacionado o asociado con el actor que tiene que demandarla.

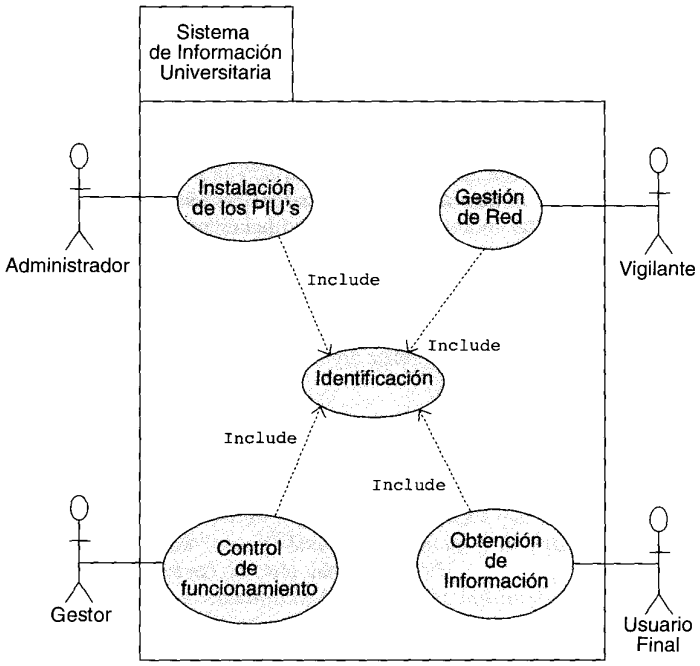


Figura 5.6:*Casos de uso del "sistema de información universitaria".*

Podemos observar con la descripción del problema que todos los actores van a tener la tarea de su idenificación previamente a la realización de cualquier tarea, con lo cual utilizaremos la relación de *include* entre la nueva funcionalidad de *Identificación* y el resto. Con ello indicamos explícitamente que para realizar cualquier operación en el sistema es necesario la identificación.

Al examinar con posterioridad el enunciado observamos que existe una serie de funcionalidades que no habíamos detectado y que mostramos en la Figura 5.7. Identificamos que la operación de instalación de PIU's tiene embebido las operaciones de Instalación de PIU existente y/o la Instalación de nuevo PIU.

Para representar esta relación entre las distintas funcionalidades que deben existir empleamos la relación de *extend* entre el caso de uso *Instalación de los PIU's y* los casos de uso *Instalación PIU existente* e *Instalación nuevo PIU.* Con ello reflejamos la semántica que nos proporciona la descripción del problema.

De forma análoga sucede con la operación de Control de Funcionamiento. Observamos que esta operación supone la realización o no de la función de Determinar las acciones por mal funcionamiento, Realizar las acciones correctivas y Actualizar los parámetros de los PIU.

Representamos pues estos casos de uso con una relación *extend* entre el caso de uso *Control de funcionamiento y* los casos de uso *Determinar Acciones Mal Funcionamiento, Actualizar Parámetros PIU y Realizar Acciones Correctivas.* De esa forma reflejamos el carácter de opcionalidad al realizar la función de control de funcionamiento.

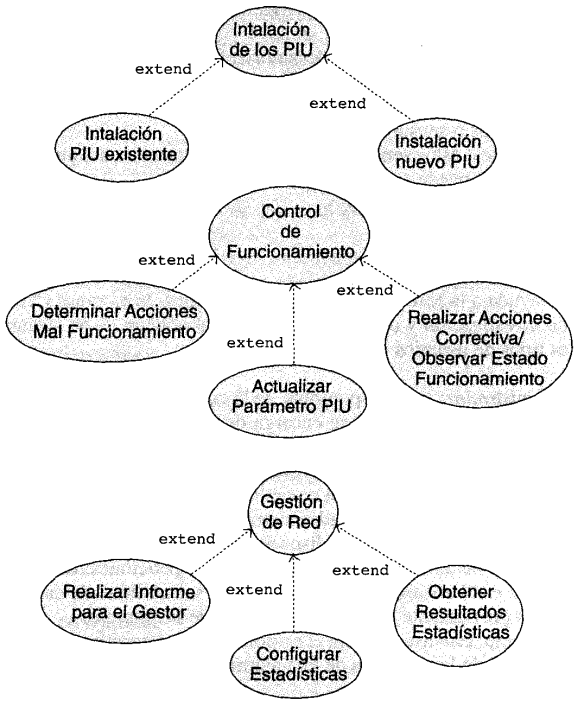


Figura 5.7: *Relaciones entre los casos de uso del "sistema de información universitaria".*

Finalmente también sucede lo mismo con la funcionalidad de *Gestión de Red*, ya que supone las operaciones de *Realizar Informe, Configurar Estadísticas y Obtener Resultados de Estadísticas.* Según el enunciado, estas operaciones pueden realizarse en determinados momentos, lo que supone que para relacionar los distintos casos de uso que conforman cada una de estas operaciones es necesario utilizar la relación de *extend* entre el caso de uso *Gestión de Red y* los otros tres.

Tema 6. Diagrama de Clases. Ejercicios Resueltos

Ejercicio 1. Animales de la casa

Enunciado

Diseñar una aplicación orientada a objetos que describa la siguiente situación:

> En una casa viven cinco animales: una ballena llamada "Moby Dick", que no dice nada; un perro fiero llamado "Caín", que dice "Grrr"; un perro manso llamado "Abel", que dice "Guau"; un pingüino llamado "Adela" que no dice nada, y un loro que dice "Lorito bonito", "Pretty Polly" y "Viva mi dueño".

Especificar la jerarquía de herencia, las clases, los atributos y los métodos de cada clase.

Solución

El primero de los pasos que debemos realizar para obtener el diagrama de clases del problema consiste en extraer todos los sustantivos que aparecen en el enunciado. La lista de sustantivos escritos, en singular, y que pueden ser, por tanto, una clase es la siguiente:

Casa Perro Animal Ballena Pingüino Loro

Tras identificar todas las posibles clases que se extraen de forma directa de enunciado pasamos a eliminar aquellos que no aportan nada para modelar el problema.

Eliminamos, por tanto, la clase Casa, ya que hace referencia al lugar en el que se describe la situación y no aporta nada.

Tras quedarnos con las clases candidatas identificamos los atributos de las clases. Para ello observamos los posibles valores que una propiedad de una clase puede tomar, el rango de los valores o una regla que enuncia a todos los posibles valores.

De la única clase que podemos extraer un atributo es de la clase Perro y el nombre del atributo tiene que reflejar el hecho que un Perro sea fiero o manso, así que definimos el atributo Agresividad que puede contener los valores de fiero o manso. El atributo será privado y por ello aparece en el diagrama el símbolo de "-" a la izquierda del nombre.

A continuación recogemos las operaciones de una clase, en nuestro enunciado se hace referencia a que un perro hace `guau' y un loro repite palabras varias veces, así que podemos deducir que el Perro tendrá una operación Ladrar y el Loro una operación Hablar. A diferencia que en el caso del atributo el símbolo que aparece a la izquierda de los nombres de las operaciones es un "+" indicando que son públicas.

Finalmente recogemos la herencia existente entre las clases en la Figura 6.1.

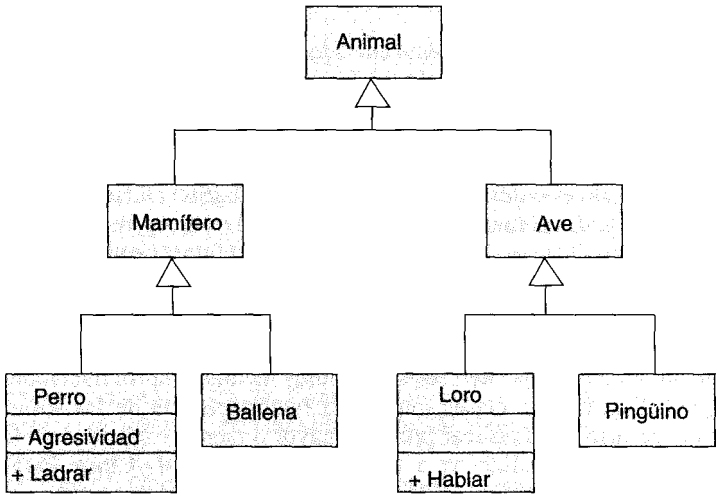


Figura 6.1*: Diagrama de clase.*

Ejercicio 2. Reserva de vuelos

Enunciado

El sistema de reserva de vuelos es un sistema que permite al usuario hacer consultas y reservas de vuelos, además de poder comprar los billetes aéreos de forma remota, sin la necesidad de recurrir a un agente de viajes humano. Se desea que el sistema de reservas sea accesible a través de la World Wide Web.

El sistema actualmente tiene un Terminal de Servicio de Reserva (formado por un ratón Genius, teclado IBM y monitor Sony) en donde se presenta un mensaje de bienvenida describiendo los servicios ofrecidos junto con la opción para registrarse por primera vez, o si ya se está registrado, poder utilizar el sistema de reserva de vuelos. Este acceso se da por medio de la inserción de un login previamente especificado (dirección de correo electrónico del usuario) y una contraseña previamente escogida y que debe validarse.

Una vez registrado el usuario, y después de haberse validado el registro y contraseña del usuario, se pueden seleccionar las siguientes actividades:

• Consulta de vuelos.

• Reserva de vuelos.

• Compra de billetes.

La consulta de vuelos se puede hacer de tres maneras diferentes:

• Horarios de Vuelos.

• Tarifas de Vuelos.

• Información de Vuelo

La consulta según horario muestra los horarios de las diferentes aerolíneas que dan servicio entre dos ciudades. La consulta según tarifas muestra los diferentes vuelos entre dos ciudades ordenados por su costo. La información de vuelos se utiliza principalmente para consultar el estado de algún vuelo, incluyendo información de si existen asientos disponibles y, en el caso de un vuelo para el mismo día, si éste está en hora. Se pueden incluir preferencias en las búsquedas, como fecha y horario deseado, categoría de asiento, aerolínea deseada y si se desean sólo vuelos directos.

La reserva de vuelo permite al cliente hacer una reserva para un vuelo particular, especificando la fecha y horario, bajo una tarifa establecida. Es posible reservar un itinerario compuesto de múltiples vuelos, para uno o más pasajeros, además de poder reservar asientos.

La compra permite al cliente, dada una reserva de vuelo previa y una tarjeta de crédito válida, adquirir los billetes aéreos.

Los billetes serán posteriormente enviados al cliente, o estarán listos para ser recogidos en el mostrador del aeropuerto antes de la salida del primer vuelo.

Es necesario estar previamente registrado con un número de tarjeta de crédito válida para poder hacer compras de billetes, o bien proveerla en el momento de la compra.

Además de los servicios de vuelo, el usuario podrá en cualquier momento leer, modificar o cancelar su propio registro, todo esto después de haber sido el usuario validado en el sistema.

Finalmente se nos comenta que existe un operador encargado del mantenimiento del sistema, pero no se describirá su labor, dejando dichas descripciones para futuras entrevistas.

Solución

Se presentan los diagramas de clases obtenidos mediante aproximaciones sucesivas. El proceso de construcción del diagrama de clases implica la realimentación de las soluciones conseguidas tantas veces como sea necesario, sin implicar por ello mayor o menor capacidad de los analistas.

El primer paso a realizar va a ser la Identificación de Clase. Para ello se subrayan todos los sustantivos en la descripción del problema, identificándose los siguientes sustantivos, correspondientes a las clases candidatas (excluyendo repeticiones y manteniendo todo en singular):

Sistema de reserva de vuelo	*Hora*	*Compra de billetes*
Sistema	*Preferencia*	*Horario de vuelos*
Usuario	*Búsqueda*	*Tarifa de vuelos*
Consulta	*Fecha*	*Información de vuelo*
Reserva	*Horario*	*Horario*
Vuelo	*Agente de viajes humano*	*Aerolínea*
Billete aéreo	*Sistema de reservas*	*Ciudad*
Login	*World wide web*	*Tarifa*
Dirección de correo electrónico	*TSR*	*Costo*
Contraseña	*Ratón*	*Estado*
Registro	*Teclado*	*Información*
Actividad	*Monitor*	*Categoría de asiento*
Consulta de vuelos	*Mensaje de bienvenida*	*Vuelo directo*
Reserva de vuelos	*Servicios*	*Cliente*
Asiento	*Opción*	*Itinerario*
Día	*Acceso*	*Pasajero*

Tarjeta de crédito	*Mostrador del aeropuerto*	*Compra*
Billete	*Número de tarjeta de crédito*	*Operador*
		Entrevista

El segundo paso que vamos a realizar va a ser la Selección de Clases. En este proceso de selección vamos a eliminar las clases innecesarias, para ello vamos a explicar el desarrollo completo de algunas clases y sus consideraciones de elección, siendo el resto deducibles de forma inmediata.

? *Clases redundantes: Cliente y Usuario. Usuario* puede ser más descriptivo para una aplicación informática. En el caso del Sistema de Reserva, *Cliente* es más descriptivo y se mantiene. Los sustantivos eliminados se listan a continuación con los sustantivos preferidos entre paréntesis:

Consulta de vuelo (consulta).

Reserva de vuelo (reserva).

Compra de billete (compra).

Sistema de reservas de vuelo (sistema de reservas).

Billete (billete aéreo).

Costo (tarifa).

Tarifa de vuelo (tarifa).

Vuelo directo (vuelo).

Login (email).

Horario (hora).

Fecha (día).

Dirección de correo electrónico (email).

? *Clases irrelevantes: Mostrador del Aeropuerto, Agente de Viajes Humano y Billete Aéreo.*

? *Clases imprecisas: Sistema, Servicios, Actividad, Preferencia, Búsqueda, Información, Estado, Opción, Acceso, Itinerario,* son clases imprecisas. Durante la introducción de herencia puede que sea necesario una clase para compartir aspectos comunes a ambas clases.

? *Nombres de clases: aeropuerto* en lugar de *ciudad.*

? *Clases que son atributos: Número de Tarjeta de Crédito* es un atributo de *Tarjeta de Crédito,Categoría de Asiento (asiento), información de vuelo (vuelo) y horario de vuelo (vuelo).*

? *Clases que son operaciones: Consulta, Compra, Reserva.*

? *Clases de interfaces de usuario: mensaje de bienvenida, hoja principal.*

? *Clases del sistema completo: Sistema de reserva.*

? *Clases actores: Cliente, Operador* (opcional, ya que es una ampliación del sistema).

A continuación tenemos cuáles son las clases candidatas de nuestro sistema a analizar:

Reserva	*asiento*	*contraseña*
Vuelo	*día*	*email*
Aerolínea	*hora*	*registro*
Aeropuerto	*pasajero*	*TSR*
Tarifa	*tarjeta de crédito*	

Después de haber identificado y seleccionado las clases, se construye un primer diagrama de clases para el dominio del problema (ver Figura 6.2).

Como podemos observar, se han eliminado aquellas clases candidatas que son atributos.

El siguiente paso que realizaremos será la **Identificación de las Relaciones**. Inicialmente se muestran las relaciones básicas existentes entre las diferentes clases del sistema. Para ello identificamos las siguientes frases:

Reserva de vuelos.

Asientos en un vuelo.

Fecha y horario de vuelo.

Aerolínea deseada.

Tarifa de vuelo.

Itinerario de vuelos.

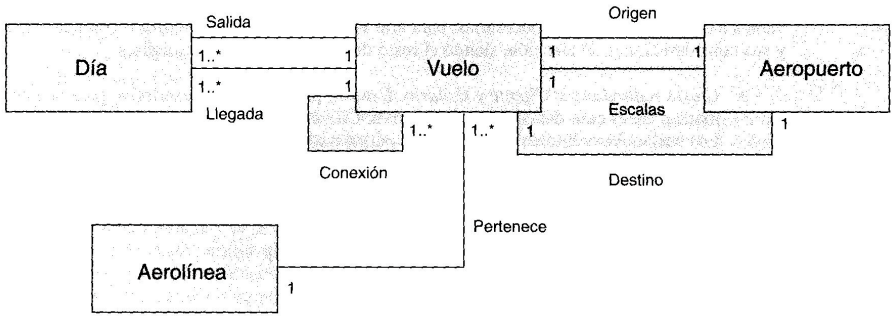


Figura 6.2: *Primera aproximación al diagrama de clases*

Reescribimos las frases para así obtener las candidatas:

El vuelo contiene reservas. El vuelo contiene asientos. El vuelo tiene día y hora.

El vuelo pertenece a una aerolínea. El vuelo tiene tarifas.

El vuelo se compone de un itinerario. El pasajero tiene reservas.

El pasajero posee una tarjeta de crédito.

Tras haber identificado y seleccionado las asociaciones, se construye un diagrama de clases con las asociaciones, los roles y la multiplicidad quedando el siguiente diagrama de clases que se muestra en la Figura 6.2.

El **Vuelo** se denomina por medio de un número, tiene como origen un aeropuerto en una ciudad y tiene como destino un aeropuerto de otra ciudad. Un vuelo puede tener múltiples escalas y múltiples vuelos, se relacionan por medio de conexiones. El vuelo pertenece a una **aerolínea** y puede operar varios **días** a la semana teniendo un horario de salida y otro de llegada.

El **Aeropuerto** sirve como origen, destino y escalas de un vuelo. El aeropuerto se encuentra en una ciudad de un país determinado.

Se identifica una clase adicional, como **Avión**, y las relaciones básicas existentes entre las clases Aerolínea, Avión, Tarifa, Asiento y Vuelo.

La **Aerolínea** provee servicio de múltiples vuelos entre diferentes ciudades bajo diferentes horarios. La aerolínea se identifica por un nombre.

Un **vuelo** en una fecha determinada se hace en un tipo de **avión** particular. El tipo de avión define la cantidad máxima de pasajeros que pueden viajar en ese vuelo para esa fecha.

Los diferentes vuelos tienen múltiples **tarifas** para compra de billete, variando según la clase de billete, si son de ida o de ida y vuelta, y dependiendo de las diversas restricciones y ofertas existentes.

En las reservas de vuelos se puede incluir una solicitud de asignación de **asiento**, especificando preferencias como pasillo o ventana. El número de asientos disponibles en un vuelo particular depende del tipo de avión que opere ese día. En el diagrama de la Figura 6.3 se muestran las relaciones entre las clases descritas anteriormente.

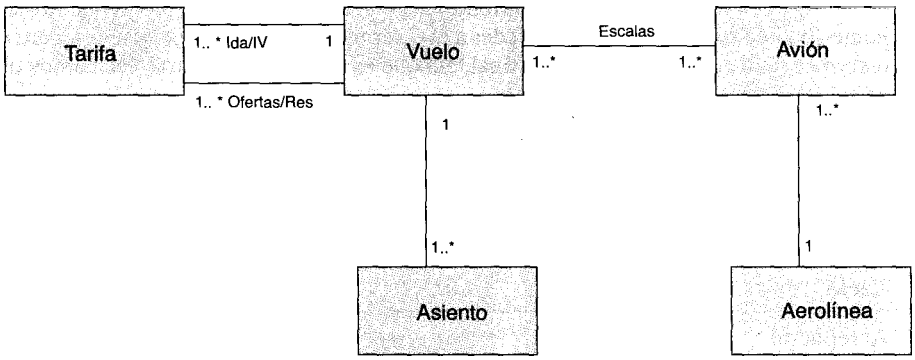


Figura 6.3: *Segunda aproximación del diagrama de clases*

Otras de las relaciones que se encuentran son las siguientes:

? El horario de un vuelo se define según los **días** en que opera.

? El horario de un vuelo se determina por su **hora** de salida y hora de llegada durante los **días** que opera. Así pues, el diagrama resultante de esta asociación entre la clase Día y Hora se muestra en la Figura 6.4.

? Para poder tomar un vuelo es necesario contar con una **reserva** previa, la cual debe pagarse antes de una fecha límite, que puede ser el propio día del vuelo. Una reserva puede hacerse para múltiples vuelos y múltiples pasajeros. La reserva cuenta con una clave que identifica un registro de reserva particular.

Figura 6.4: *Relación entre las clases Hora y Día*

Se identifica una clase adicional llamada Pago, que consta de información sobre la cantidad, fecha y tipo de transacción.

Por razones de seguridad, los pagos de billete se hacen mediante tarjeta de crédito. El diagrama que muestra las relaciones anteriores es el de la Figura 6.5.

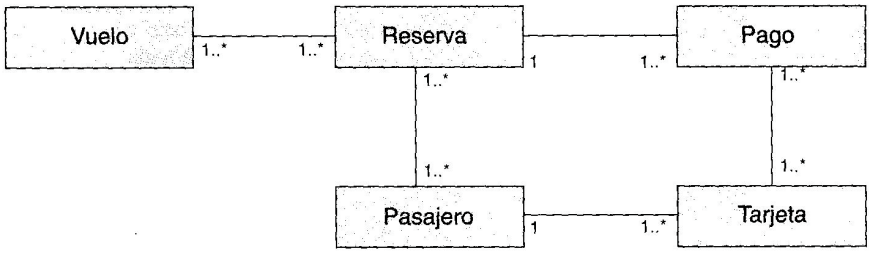


Figura 6.5: *Tercera aproximación al diagrama de clases.*

Finalmente identificamos los atributos según la descripción del problema. Los atributos de las clases los hemos podido obtener antes de proceder a la determinación de las asociaciones, pero en este caso nos ha parecido más fácil su representación al final, aunque es obvio que mientras estábamos obteniendo las relaciones extraíamos los atributos.

Así pues, tenemos los siguientes atributos asociados a cada clase:

Nombre de la Clase	**Nombre del Atributo**
Vuelo	Número
Aeropuerto	Ciudad País
Aerolínea	Nombre
Avión	Compañía
	Tipo N.°
	Pasajeros
Asiento	Fila
	Letra
Tarifa	Clase
	Precio
	Impuestos
Reserva	Clave
	Costo
	Total

Pago	Tipo
	Fecha
	Cantidad
Tarjeta	Nombre
	Número
	Tipo
	Fecha
	Vencimiento
Pasajero	Nombre
	Dirección
	Ciudad
	País
	Código Postal
	Teléfono Casa
	Teléfono Oficina
	Fax
	Email

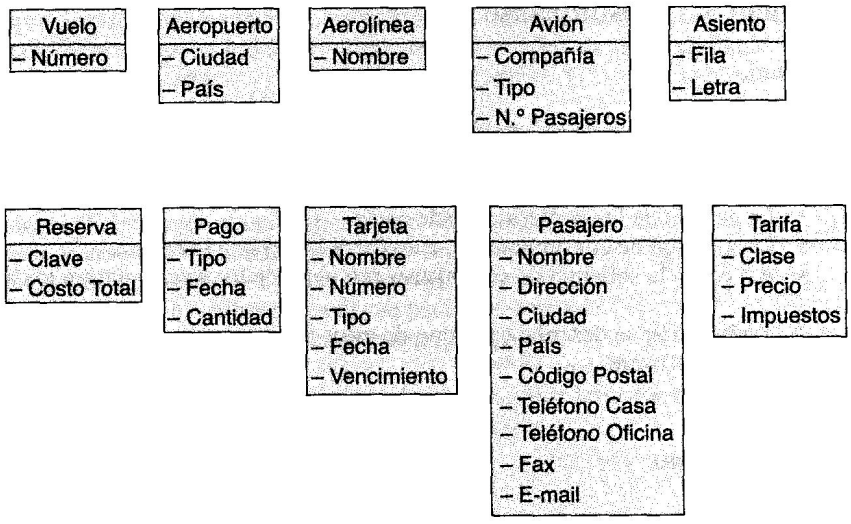


Figura 6.6: *Atributos de las clases identificadas.*

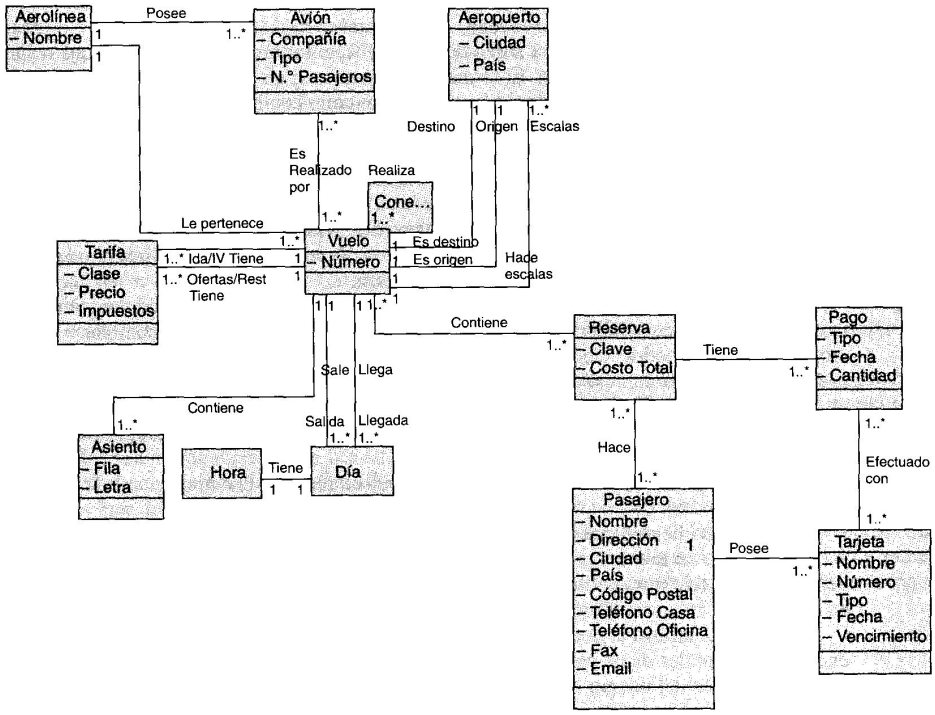


Figura 6.7: *Versión final del diagrama de clases*

En la Figura 6.6 se muestra las clases con sus atributos y en la Figura 6.7 se muestra el diagrama completo final con todas las clases y sus relaciones.

Ejercicio 3. Restaurante

Enunciado

?El dueño de una cadena de restaurantes de Madrid quiere que se hagan de forma automática:

?Las reservas de las mesas de sus restaurantes. La gestión de los pedidos de cada mesa.

?La solicitud de consumiciones, comidas y bebidas, a la cocina.

?Así como la solicitud de suministros por parte de los restaurantes a los almacenes.

A continuación se describe cada uno de estos procesos que se quieren automatizar, mediante el uso de una aplicación software.

Reservas de mesas

Los clientes de los restaurantes pueden llamar por teléfono para reservar una mesa, pero lo que se está intentando poner de moda es el uso de unos terminales punto de reserva (TPR) ubicados en la calle. La ventaja que tiene el uso de estos terminales es la posibilidad de elegir la mesa en función de su ubicación dentro del restaurante, cosa que no se puede hacer por teléfono.

Todos los TPR son de la cadena de restaurantes, aunque cabe la posibilidad de que en un futuro distintas cadenas de restaurantes puedan ofrecer sus servicios a través de estos terminales. Hoy por hoy sólo se podrán elegir restaurantes de esta cadena de restaurantes.

Cuando un cliente se conecta a uno de estos TPR, el terminal le pregunta en qué restaurante quiere realizar la reserva, qué día y la hora. El terminal comprueba si en el restaurante especificado hay alguna mesa libre a esa hora. Si es así, muestra el plano del restaurante con las mesas que hay libres.

Las mesas están separadas en mesas de fumador, marcadas con la F, y de no fumador, marcadas con NE Además, cada mesa lleva un indicador con el número de personas para el que está pensada dicha mesa.

El usuario selecciona una mesa e indica el número de personas que van a ocuparla; si todo está bien, el terminal pide al usuario que indique el nombre con el cual desea realizar la reserva, el usuario se lo indica y el terminal le da un ticket indicando el día, la hora, la mesa y el nombre con el que ha reservado la mesa.

Si el cliente llega al restaurante veinte minutos después de la hora de reserva de la mesa, el sistema se encargará automáticamente de dejar libre dicha mesa.

Si no hay mesas libres a la hora indicada por el usuario, el TPR se lo comunica al cliente, dándole además la posibilidad de solicitar al sistema sugerencias sobre restaurantes disponibles a la hora y en el día solicitado. El usuario podrá seleccionar alguno, en cuyo caso el procedimiento es el mismo que para el caso de la reserva normal, exceptuando que el TPR ya tiene ciertos datos del cliente.

Si lo que ocurre es que sí hay mesas, pero el cliente no encuentra ninguna mesa que le satisfaga a la hora a la que desea la reserva, puede solicitar al sistema que le indique otro restaurante de la cadena que también tenga mesas libres a esa hora.

Si en cualquiera de los casos el usuario cambia de idea, basta con que cancele en cualquier momento la operación.

Cuando un cliente llega a uno de los restaurantes de la cadena, se le pregunta si tiene reserva o no.

En el caso en que tenga reserva, bastará con que presente el ticket, si la hora de reserva no supera en veinte minutos a la hora de llegada al restaurante, la mesa pasa de estar *reservada* a *ocupada* y se les sienta en el lugar que les corresponde.

Si por el contrario la hora de llegada supera en veinte minutos a la hora de reserva, el sistema se habrá encargado de anular dicha reserva, de modo que la mesa haya quedado libre para otro posible cliente; por tanto, se les trata del mismo modo que si no tuvieran reserva. En ese caso el encargado, en ese momento de las reservas, solicita al sistema que le muestre las mesas libres para ese momento; si hay mesas libres, le pregunta al usuario si quiere mesa de fumador o de no fumador y cuántas personas son, el usuario se lo dice y en caso de que haya mesa libre, el encargado les sienta. Si no hay mesa, el encargado le debe pedir al sistema el tiempo aproximado para que quede libre la próxima mesa de las características de la mesa solicitada. Esto podrá calcularlo el sistema a través del estado en que se encuentran las distintas mesas en un determinado momento, estos estados son:

Libre: si nadie la ha reservado.

Reservada: si alguien ha hecho una reserva.

Ocupada: si los comensales están ya a la mesa.

Pidiendo: si el camarero está recogiendo el pedido de esa mesa.

En espera de comida: si están esperando que se les sirva.

Servidos: si los comensales ya tienen la comida en la mesa. Esperando cuenta: si los comensales hayan pedido la cuenta.

Pagando: si los comensales ya tienen la cuenta en la mesa.

Además, si no hay mesas libres y el cliente lo desea, se le debe informar de otro/s restaurante de la cadena que sí tenga mesas libres.

Pedidos

Una vez que los clientes están a la mesa, los camareros les dan la carta y esperan que pidan. Los camareros tienen unos dispositivos que controlan una parte del sistema, el de los pedidos en cada mesa.

Esta parte del sistema está a la espera de que el camarero introduzca un número de mesa.

Cuando el camarero introduce el número de la mesa que va a pedir, se graba automáticamente la hora del pedido y la mesa que lo está haciendo. Los clientes pueden pedir tanto comidas como bebidas, ambas se consideran consumiciones. Cada tipo de consumición tiene un código que será lo que el camarero introduzca en el sistema.

Si un cliente quiere saber los ingredientes de un determinado plato se lo puede preguntar al camarero, el cual, a su vez, lo consulta al sistema tecleando el código de la consumición seguido del símbolo de interrogación.

El pedido de cada mesa se va componiendo de líneas de pedido donde cada línea de pedido es una consumición. Es decir, si se piden tres platos de pasta y dos cervezas, el pedido tendrá cinco líneas de pedido.

El camarero introduce por cada consumición el código de ésta y pulsa aceptar; antes de poder volver a introducir un código de consumición, el sistema debe ser capaz de comprobar que hay ingredientes necesarios para satisfacer dicha petición de consumición. Si no fuera el caso, es decir, si no se pudiera completar la consumición por falta de uno o varios ingredientes, el camarero indicará al cliente que no es posible para que pida otra cosa. Por supuesto, al detectarse esta situación se debe informar al almacén de que reponga cada uno de los ingredientes o bebidas que faltan.

Una vez que los comensales terminan de pedir, el camarero cierra temporalmente la nota, es decir, pulsa fin, mientras no le pidan nada más y la mesa pasa a estar en estado de "Esperar comida". Automáticamente el sistema avisa en cocina que hay un nuevo pedido en una mesa determinada. En este momento se recorre cada línea del pedido, de nuevo, para ir a su vez recorriendo los ingredientes de cada consumición y disminuir la cantidad que se tiene de un determinado producto en cocina, de modo que si la cantidad del producto disminuye por debajo del umbral establecido para ese alimento se pida automáticamente a almacén.

El encargado de la cocina observa cuando llega un nuevo pedido y se lo indica a los cocineros. Cuando los platos están listos el encargado de cocina establece el pedido de esa mesa como cocinado y manda un mensaje al control del camarero para que recoja el pedido de la mesa indicada, el camarero lo recoge para llevarlo a la mesa que corresponde e indica que esa mesa está servida.

Control de Ingredientes

Además, como ya señalábamos antes, desde la cocina también se lleva el control de los ingredientes, como se sabe exactamente los ingredientes de cada plato, una vez se ha preparado la/s bandejas que contienen el pedido de una mesa, se indica al sistema que los ingredientes que contenían esos platos o consumiciones han disminuido, de modo que cuando rebasan el mínimo indispensable en cocina, el sistema avisa automáticamente para que repongan desde almacén.

Pago y liberación de mesa

Cuando los comensales han terminado, piden al camarero la nota, momento en el cual el camarero cierra definitivamente el pedido de esa mesa y establece el estado de la mesa como esperando nota. El camarero ordena que se imprima la nota que

está compuesta por cada una de las líneas de pedido. Una vez está impresa se la pasa a los clientes y éstos depositan bien el dinero en efectivo o una tarjeta. El camarero se va a la caja central e indica que esa mesa está pagando, vuelve con la nota cobrada y establece la mesa como libre.

En la Figura 6.8 se pueden observar los elementos a gestionar en el sistema objeto de estudio.

A partir de la información que nos han proporcionado, deberemos realizar el análisis orientado a objetos que se propone utilizando la técnica de los Diagramas de Clases.

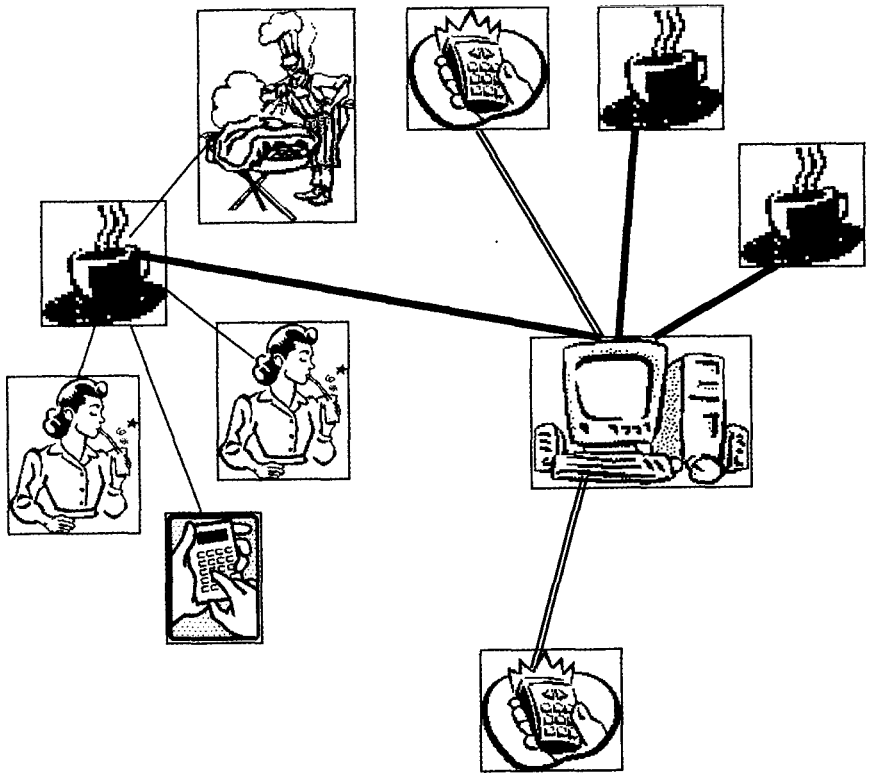


Figura 6.8: *Restaurante.*

Solución

A continuación se muestra el diagrama de clases en el que se modela la vista estática del sistema.Se presentan los diagramas de clases obtenidos mediante aproximaciones sucesivas. El proceso de construcción del diagrama de clases implica la realimentación de las soluciones conseguidas tantas veces como sea necesario, sin implicar por ello mayor o menor capacidad de los analistas.

El primer paso a realizar va a ser la Identificación de Clases.

Para ello se subrayan todos los sustantivos en la descripción del problema, identificándose los siguientes sustantivos, correspondientes a las clases candidatas (excluyendo repeticiones y manteniendo todo en singular):

Dueño	Plato	Comensal
Cadena restaurante	Cliente	Camarero
Reserva	Teléfono	Carta
Mesa	Terminal punto reserva	Dispositivo
Restaurante	Servicio	Consumición
Pedido	Día	Código
Solicitud consumición	Hora	Ingredientes
Bebida	Plano	Línea pedido

Una vez extraídos todos los nombres de la especificación del enunciado se realiza la eliminación de aquellas posibles clases que son innecesarias, es decir, que son redundantes o irrelevantes para nuestro problema. .

Eliminamos las siguientes clases redundantes, mostrando entre paréntesis el nombre de la clase que mantenemos.

?•Cadena restaurante (restaurante).

?• Cerveza (bebida).

?• Producto (consumición).

A continuación eliminamos aquellas clases que consideramos que son irrelevantes. Hay que recordar que aunque inicialmente podamos eliminar alguna clase, la realimentación de la solución que vamos obteniendo nos puede llevar posteriormente a replantear su inclusión.

Clases irrelevantes son:

? Dueño, encargado.

? Solicitud consumición y solicitud suministro, ya que representan a una operación.

? Cocina, almacén, cantidad, plato, teléfono, terminal punto reserva, plano.

? Número de persona, ya que constituirá un atributo.

? Sistema, ya que afecta a aspectos de implantación.

? Operación, ya que representa una acción en el sistema a modelar.

? Carta, nota, tiempo aproximado.

Así pues, nos quedamos con las siguientes clases: Cliente, Reserva, Mesa, Pedido, Restaurante, Consumición e Ingrediente.

Identificamos las relaciones existentes en cada clase observando las expresiones verbales existentes en el enunciado. Las principales asociaciones identificadas son las siguientes:

? Un Cliente realiza la reserva de una o varias Mesas y una Mesa puede ser reservada por varios Clientes.

? La clase Reserva es una clase asociación, ya que el contexto de su existencia está dado precisamente por la relación entre las clases Cliente y Mesa. Tenemos que resaltar que al considerar esta semántica, estamos añadiendo la restricción de que un Cliente no puede mantener varias reservas activas de la misma mesa en fechas distintas.

? Una Mesa puede tener un Pedido o no tener ningún Pedido.

? Una Mesa pertenece a un Restaurante y un Restaurante tiene muchas mesas.

? El pedido de cada mesa se va componiendo de líneas de pedido donde cada línea de pedido es una consumición. Así pues, la clase Pedido está formado por Consumiciones, con lo que la relación que los asocia es una relación de agregación.

Por otra parte, ya que una Consumición puede ser una Bebida o una Comida, la relación existente entre estas tres clases es la de herencia entre la superclase Consumición y las subclases Bebida y Comida.

Finalmente como una Consumición está formada por una serie de Ingredientes, ya sean pertenecientes a la Bebida o a la Comida pedida, existe una relación de agregación entre la clase Consumición y la clase Ingrediente.

En la Figura 6.9 se muestran las relaciones entre las clases.

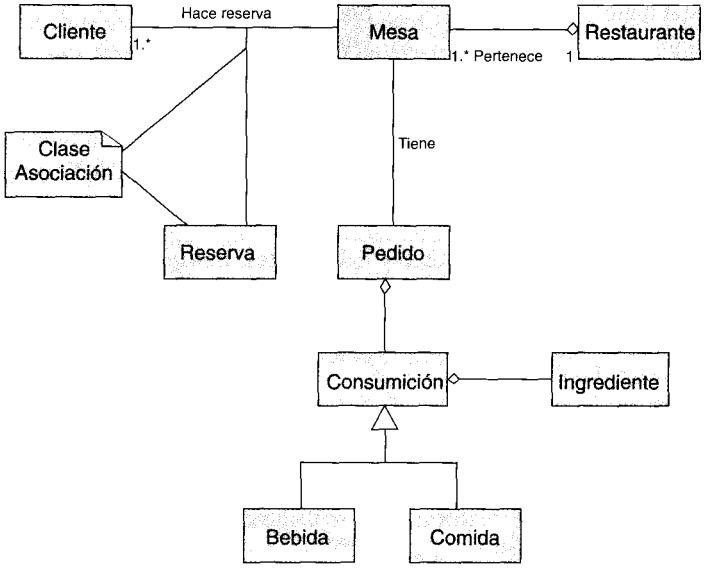


Figura 6.9: *Clases y relaciones entre ellas.*

Finalmente identificamos los atributos según la descripción del problema. Así pues, tenemos los siguientes atributos asociados a cada clase:

Nombre de la Clase	Nombre del Atributo
Cliente	Nombre
Mesa	Número_mesa Fumador Número_personas Estado Ubicación
Restaurante	Nombre Mapa Teléfono
Reserva	Fecha Hora Número__personas
Pedido	Hora
Consumición	Código Precio
Ingrediente	Cantidad disponible Cantidad Mínima

En la figura 6.10 se muestra el diagrama completo final con todas las clases, sus relaciones y sus atributos.

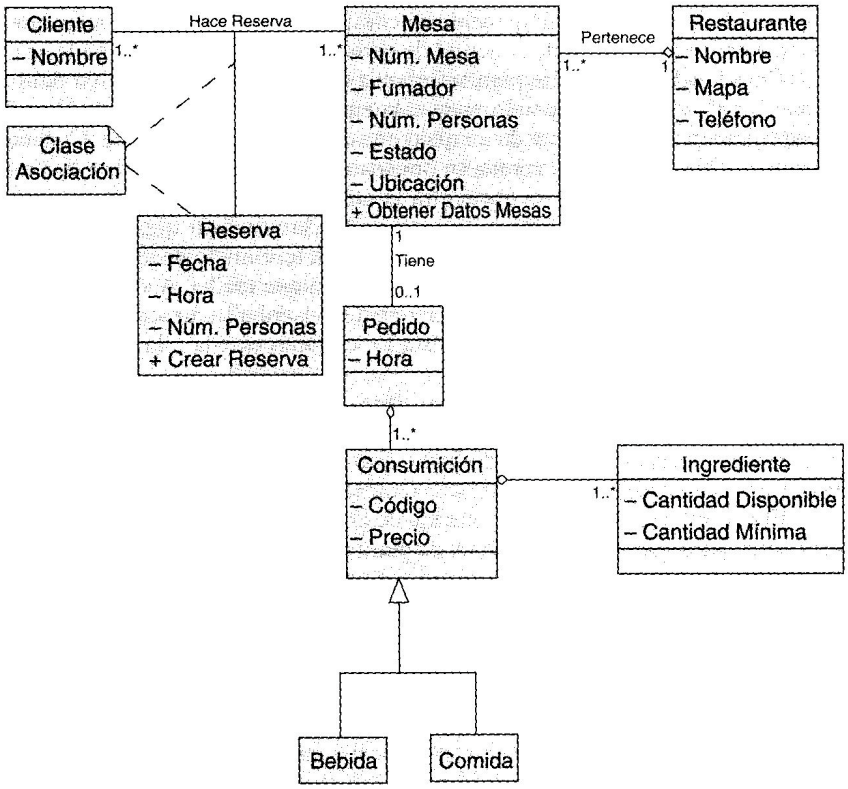


Figura 6.10: *Diagrama de clases completo*

Ejercicio 4. Parque de atracciones

Enunciado

La empresa DIVERTIMENTO, S. A., tiene varios parques de atracciones repartidos por la geografía española. Lo que más preocupa a esta empresa es la seguridad en algunas de las atracciones, ya que un error mecánico podría producir daños materiales y humanos que plantearían serios problemas para la empresa

Hoy por hoy sólo es posible detectar fallos en las atracciones, cuando los operarios encargados realizan actividades de mantenimiento.

La empresa quiere informatizar sus parques de atracciones y para ello ya ha decidido poner en marcha un proyecto piloto cuyo objetivo será el de dotar a uno de sus parques de atracciones de un sistema de detección automática de fallos en las atracciones.

En un primer momento se va a preparar el sistema para gestionar la noria y la montaña rusa. La noria tiene una serie de vehículos dotados cada uno de ellos de un detector gracias al cual se sabe en cada momento si el vehículo está

suficientemente bien anclado a la estructura metálica de la noria. Si en un momento determinado se detectara pérdida de anclaje, el correspondiente vehículo se lo comunicaría a la Central Receptora de Averías (CRA) y también a la atracción de la que forma parte dicho vehículo, así en la próxima parada de dicha atracción se tendrá constancia de que uno de sus vehículos ha solicitado revisión. Por su parte, en la montaña rusa cada coche está dotado de igual modo de un detector de anclaje con el coche que lleva detrás (en el caso de llevarlo). Cada coche detecta si existe suficiente anclaje con el coche posterior y en caso de falta de anclaje avisa a la CRA y a la atracción, en este caso la montaña rusa.

Cuando la CRA recibe un aviso, en el que se le indica el vehículo o coche con posible avería y la atracción de que se trata, busca inmediatamente un operario de mantenimiento disponible. En caso de no haber ninguno libre, informa al componente en cuestión de que su petición no puede ser satisfecha, así dicho componente emitirá una señal de solicitud de revisión hasta que su petición le sea satisfecha.

Como cada operario de mantenimiento cobra un extra en función del número de averías que atiende al mes, cada uno tiene asignado mensualmente un dispositivo gracias al cual recibe las posibles averías a atender, independientemente de en qué zona del parque se encuentre. Cuando la CRA demanda la revisión de una posible avería y encuentra un operario de mantenimiento libre le manda un mensaje indicándole la calle del parque en la que se encuentra la atracción y el número de vehículo o coche con posible avería. Automáticamente, el dispositivo del operario pasa a indicar que ese operario se encuentra ocupado atendiendo una posible avería. Cuando el operario ha terminado de supervisarla, indica a su dispositivo que ha quedado libre para la siguiente petición de avería que reciba. A su vez dicho dispositivo informa a la CRA y al componente revisado. Dicho componente avisará a su atracción de que la operación de mantenimiento solicitada ha terminado para que ésta lo tenga en cuenta a la hora de poner la atracción en marcha de nuevo.

Además, el sistema tendrá que ser capaz de contabilizar las personas que entran y salen de una atracción, con el fin de controlar dos cosas; en primer lugar, que no entren más personas de las que la atracción es capaz de albergar y, en segundo lugar, que todo el mundo abandone la atracción una vez finalizado cada viaje.

El controlador de arranque y parada de la atracción puede recibir un mensaje indicando que la atracción está llena, para que inicie las labores de puesta en marcha de la atracción; dicho mensaje puede provenir del torniquete de entrada que detecta cuando se produce la ocupación máxima de la atracción, o bien del propio operario que vigila la atracción siempre que aún no estando llena no hay más personas esperando para subir y él considera que es tiempo suficiente como para que se ponga en marcha.

Una vez que el dispositivo de parada y arranque de la atracción detecta que la atracción está detenida, le envía al torniquete de salida un mensaje para que se prepare para que la gente pase por él. El torniquete de salida sabe el número de personas que hay en la atracción gracias al torniquete de entrada, así sabe el número de personas que se tienen que bajar de la atracción.

Cuando el torniquete de salida determina que el número de personas que han abandonado la atracción es igual al número de ellas que entró, envía al torniquete de entrada un mensaje para que ponga a cero el contador de personas en la atracción y además se libere y muestre un indicador verde para que la gente pueda tomar asiento en la atracción. Si pasados cinco minutos desde que la atracción se paró el torniquete de salida no ha liberado al torniquete de entrada, es indicativo de que alguien se ha quedado dentro y es necesario entrar a buscarlo.

Cuando el torniquete de entrada recibe, del torniquete de salida, el mensaje de liberarse, primero consulta a la atracción si tiene alguna avería pendiente. Esto se reflejará en la atracción cuando uno o varios de los vehículos o coches soliciten reparación. La atracción lleva un contador de averías pendientes de manera que sólo en el caso en que este contador esté a 0 el torniquete de entrada se pondrá verde para que entren los usuarios. En caso contrario permanecerá en ámbar, indicativo de estar esperando reparación.

Solución

A continuación se muestra el diagrama de clases en el que se modela la parte estática del sistema.

Vamos a identificar las posibles clases que podemos utilizar en nuestro modelo, examinando la especificación que se ha dado en el problema y extrayendo los sustantivos existentes. Las clases candidatas que hemos seleccionado inicialmente son las siguientes:

Parque de atracciones	Geografía	Empresa
Atracción	Problema	Instalación
Operario	Actividad	Proyecto
Sistema	Noria	Montaña rusa
Vehículo	Detector	Central Receptora de Averías (CRA)
Componente	Señal	Petición

Mes	Dispositivo del operario	Mensaje
Calle	N.° de vehículo	Personas
Sala	Controlador	Torniquete
Contador	Avería	Coche

Una vez que hemos identificado las posibles clases, realizamos la selección de las clases que van a participar en el diagrama de clases. Para ello lo primero que hacemos es eliminar aquellas clases irrelevantes, ya sea porque tiene poco o nada que ver con el problema que se está modelando, porque no son específicas del entorno de la aplicación o porque son atributos de clase. Así pues, eliminamos las siguientes clases:

Parque de atracciones	Geografía	Empresa
Problema	Instalación	Actividad
Proyecto	Sistema	Detector
Componente	Señal	Petición
Mes	Mensaje	Calle
N.° de vehículo	Personas	Sala
Controlador	Contador	Avería
Daños materiales		

Eliminamos la clase Operario, ya que consideramos que el operario se relaciona con el sistema mediante la clase interfaz Dispositivo del operario.

Finalmente las clases que van a participar en nuestro modelo son las siguientes:

Atracción	Noria	Montaña rusa
Vehículo	Central Receptora de Averías (CRA)-Clase	Dispositivo de Operario
Torniquete	Coche	

Tras identificar las clases definimos cuáles son las relaciones existentes entre ellas.

Examinando de nuevo el enunciado y aplicando el proceso de generalización y especialización, identificamos la clase Torniquete y las subclases Torniquete de Entrada y Torniquete de Salida. Por tanto, existe una relación de herencia entre estas tres clases.

Entre las subclases existe una relación uno a uno entre el Torniquete de Entrada y el de Salida que permite comparar si el número de personas que salen es igual al que entran.

Una Atracción se activa por un Torniquete de Entrada y cada Torniquete de Entrada activa a una única Atracción.

De igual forma una Atracción comunica a un Torniquete de Salida que debe prepararse y un Torniquete de Salida se activa gracias a una única Atracción.

Aunque podríamos crear una relación entre Atracción y Torniquete y eliminar, por tanto, la que une a cada uno de los tipos de Torniquete, hemos preferido conservarlas para matizar que la asociación es distinta con cada uno de los tipos de Torniquete.

Ya que una Atracción puede ser una Noria o una Montaña realizando las funciones de Atracción, utilizamos la relación de herencia para relacionar a la clase Atracción con las subclases Noria y Montaña.

Por otra parte, una Noria está formada por vehículos, con lo cual existe una relación de agregación entre Noria y Vehículo. De igual forma ocurre entre Montaña rusa y Coche.

Como un Coche arrastra a otro Coche, tenemos una relación reflexiva en la clase Coche.

La Central Receptora de Avenas (CRA) mantiene a una o varias Atracciones, mientras que Atracción sólo se mantiene por una CRA.

Finalmente una CRA gestiona varios Dispositivos de operario y un Dispositivo de operario es gestionado por una única CRA.

En la Figura 6.11 se muestra el diagrama de clases, con las operaciones y atributos que hemos identificado a partir de una lectura minuciosa de la especificación del problema.

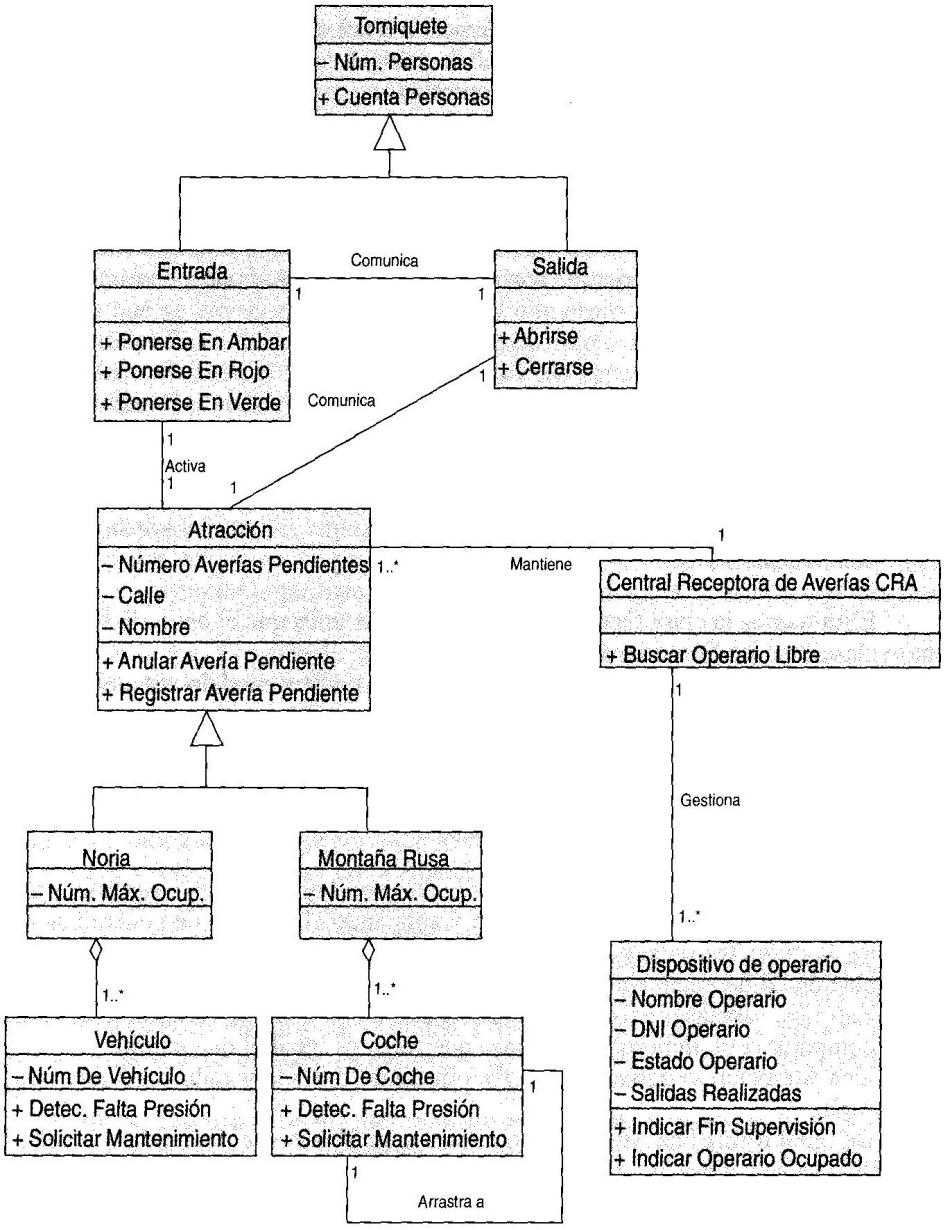


Figura 6.11: *Modelos de clases de la gestión del Parque*

Tema 7. Diagramas de Interacción. Ejercicios Resueltos

Ejercicio 1. Gestión de dietas

Enunciado

Se pretende modelar el funcionamiento de un servicio de atención médica. El hitolfase actual del proyecto es el desarrollo del MAD (Módulo Automatizado de Dietética), con él se pretende que el médico cuente con una herramienta que facilite la asignación de dietas a los pacientes. Para poder llevar a cabo sus funciones el MAD deberá poder consultar información sobre los pacientes (su historia clínica), las enfermedades y los posibles tratamientos (dietas). Para la obtención de las posibles dietas el MAD cuenta con un módulo subordinado (al que emite solicitudes) denominado DIETAS, que es el encargado de definir y prepocesar dietas para el MAD.

La operativa de trabajo utilizada para la automatización de la realización de diagnósticos y tratamientos se define en la enumeración de los siguientes pasos:

1. Un módulo no definido actualmente y denominado Gestor de Solicitudes (GS) es el encargado de solicitar un tratamiento al MAD, proporcionándole como única información el paciente a tratar.

2. El módulo de dietas (MAD) obtiene la historia clínica del paciente. La historia clínica del paciente sólo se facilita al MAD si dicho paciente está adscrito al servicio de Nutrición. En otro caso se produce una situación de excepción que se soluciona informando al MAD y éste a su vez al GS, dando de esta manera por finalizada la petición de tratamiento.

3. Para cada una de las enfermedades a tratar que el módulo MAD recibe, emite una solicitud de dieta al módulo DIETAS, incluyendo en ella todos los datos necesarios para que se lleve a cabo con éxito.

4. El módulo DIETAS para cada una de las peticiones de dieta que recibe solicita información de todas las fuentes alimentarias asociadas a los nutrientes, cuyo déficit produce la enfermedad a tratar. Que una vez recibe, le sirven para generar una dieta aconsejada, que envía al módulo de dietas (MAD).

Una vez que el módulo de dietas (MAD) recibe todas las dietas aconsejadas para todas las enfermedades para las cuales solicitó tratamiento. Las readapta teniendo en cuenta las condiciones características del caso que se está tratando y las une. Generando una dieta final verificada que es enviada al GS.

Se pide representar un diagrama de secuencia y el correspondiente diagrama de colaboración, que contemple las acciones desencadenadas por el sistema cuando el módulo MAD recibe una solicitud de tratamiento emitida por el Gestor de Solicitudes (actor) y se dan todas las condiciones para que la petición llegue a buen término.

Solución

El primer paso a dar para construir un diagrama de secuencia es identificar cuál es la actividad del sistema que se quiere representar y quién es el actor que desencadena dicha actividad.

En este caso, la actividad en cuestión es aquella a través de la cual el sistema es capaz de proporcionar la dieta más apropiada para un paciente concreto. El actor que desencadena o demanda dicha actividad es el Gestor de Solicitudes (GS), tal y como se extrae del paso uno de la operativa de trabajo descrita en el enunciado, ya que éste demanda al sistema que inicie el mecanismo que termina con la obtención de la dieta más apropiada.

A continuación se deben identificar las clases que van a participar en el desarrollo de dicha actividad. Para ello, se debe identificar una secuencia de pasos más cortos en los que se divide dicha actividad y listarlos uno tras otro. Dicho conjunto de pasos que conducen a la obtención de la dieta específica pueden aparecer diseminados en distintos puntos de la especificación del problema a resolver. En este caso los pasos son los siguientes:

1. El GS solicita tratamiento para un paciente concreto y se lo solicita al MAD (Módulo de Dietas): de este paso se deduce que se tiene un actor, el GS, que solicita al MAD un tratamiento para un paciente. Esto queda representado en la Figura 7.1, y en la Figura 7.2 con el mensaje que le envía el actor GS a la clase MAD.

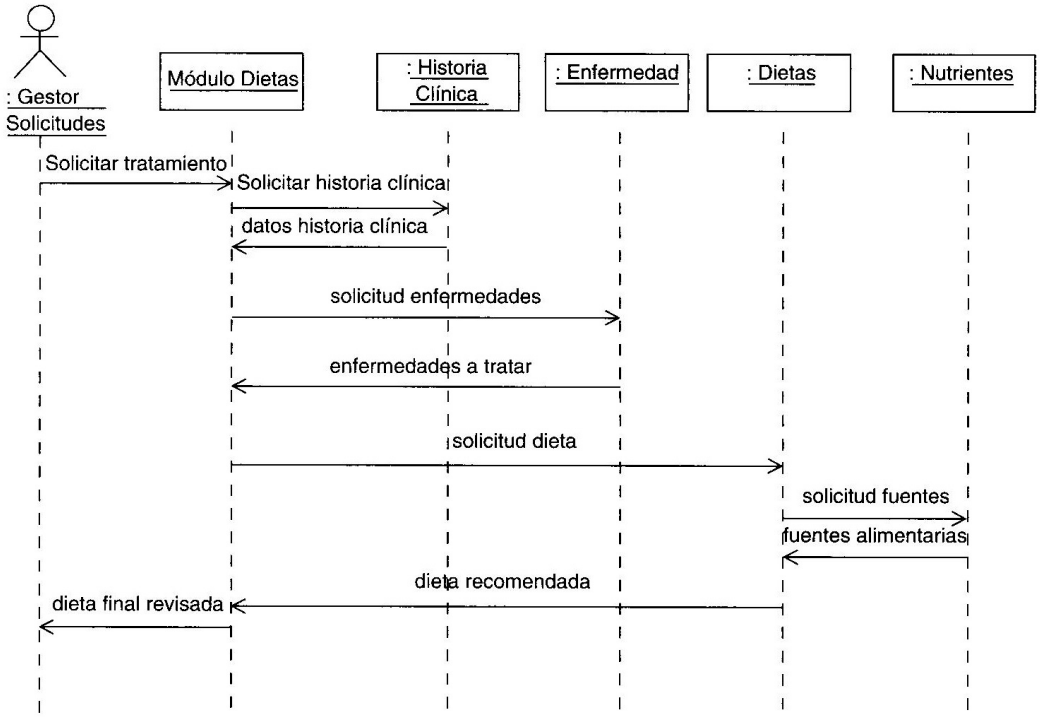


Figura 7.1: *Diagrama de secuencia de obtención de la dieta más adecuada*

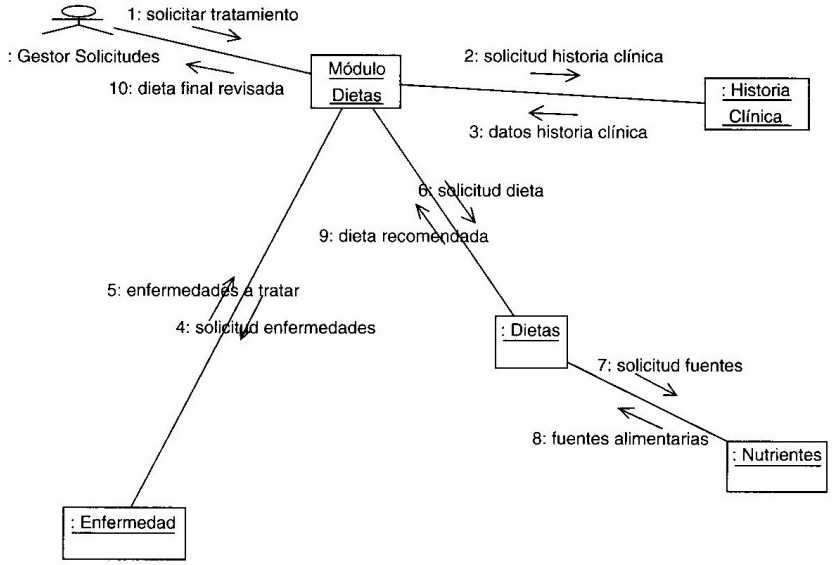


Figura 7.2: *Diagrama de colaboración de obtención de la dieta más adecuada*

2. A continuación el MAD solicita la historia clínica de ese paciente. Como la historia clínica es una clase, ya que tiene muchos datos y diferentes operaciones asociadas, esta petición se representa en la Figura 7.1, y en la Figura 7.2, con el mensaje que le envía la clase MAD a la clase Historia Clínica.

3. Si dicho paciente tiene Historia Clínica, la clase Historia Clínica le devuelve los datos correspondientes de ese paciente al MAD.

4. Por cada una de las enfermedades que aparecen en la Historia Clínica del Paciente, el MAD envía un mensaje a la clase enfermedad, para que le envíe los datos relevantes sobre dicha enfermedad. Esto queda representado en la Figura 7.1, y en la Figura 7.2, con el mensaje que le envía la clase MAD a la clase Enfermedad (solicitud enfermedades), y el mensaje que le envía la clase Enfermedad a la clase MAD (enfermedades a tratar). Ambos mensajes se repetirán tantas veces como enfermedades tenga diagnosticadas dicho paciente. Para indicar esto en UML se puede utilizar una nota, o bien comentarlo junto al diagrama a pie de página.

5. Una vez que el MAD sabe todas las enfermedades diagnosticadas al paciente en cuestión, le envía un mensaje a la clase Dietas (solicitud dietas).

6. La clase Dietas envía un mensaje a la clase nutrientes (solicitud fuentes) para identificar cuáles son los nutrientes cuyo déficit provocan cada una de las enfermedades de ese paciente. Los nutrientes le envían un mensaje a la clase dietas (fuentes alimentarias).

7. Cuando la clase Dietas ha recibido todos los mensajes de la clase Nutrientes, envía un mensaje a la clase MAD con la dieta adecuada (dieta recomendada).

8. A su vez la clase MAD, que es la encargada de conectarse con el Actor (GS), le envía un mensaje con la dieta solicitada (dieta final revisada).

La Figura 7.1 y la Figura 7.2 representan la misma secuencia anterior, la única diferencia radica en la forma de representar los datos. En la primera, la lectura se hace de arriba abajo y de izquierda a derecha. Mientras que para la segunda el punto de comienzo sigue siendo el actor (gestor de solicitudes) y la lectura se hace según un número de orden que se debe ir añadiendo a cada mensaje que se genere entre clases.

Ejercicio 2. Control de tráfico en un cruce regulado por semáforo

Enunciado

El diagrama de clases de la Figura 7.3 muestra la estructura de un controlador de tráfico empleado para regular un cruce de calles como el mostrado en la Figura 7.4. Cada calle tiene dos carriles en cada sentido, que permiten en el cruce el giro a la izquierda y a la derecha respectivamente, además de la marcha hacia delante.

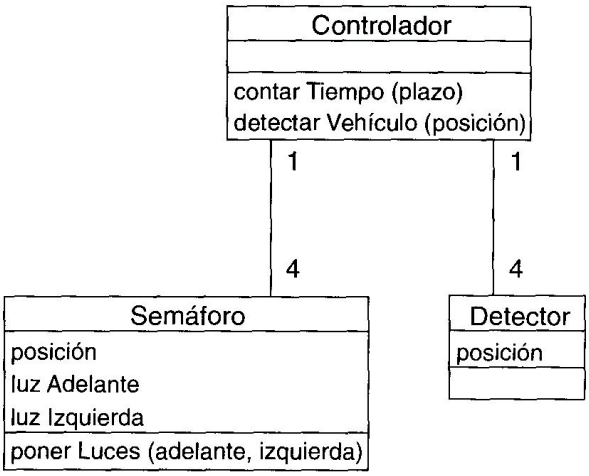


Figura 7.3: *Diagrama de clases correspondiente al control de tráfico*

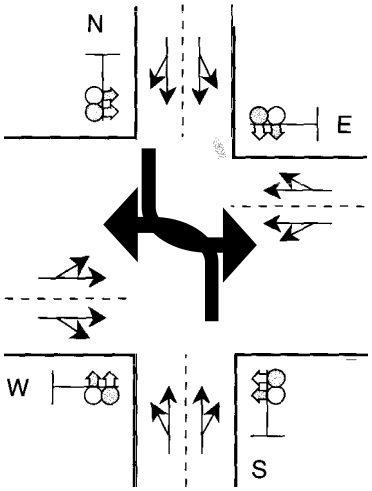


Figura: 7.4: *Cruce de calles.*

El controlador está asociado a cuatro semáforos y cuatro detectores, identificados cada uno por su posición: N (norte), E (este), S (sur) y W (oeste). El controlador alterna el tráfico en dirección N-S y de acuerdo con un temporizador interno. En cada ciclo de funcionamiento de un semáforo se permite primero la circulación hacia delante (o hacia la derecha) y a continuación el giro hacia la izquierda.

Cada semáforo tiene dos hileras verticales de luces, una para controlar la circulación hacia de la otra hacia la izquierda (siempre que se puede ir hacia delante se puede girar a la derecha, de modo que necesaria una hilera especial para permitir o prohibir el giro a la derecha). Para simplificar, en cada 1 sólo se consideran las luces roja y verde, despreciando los breves segundos que el semáforo está en á antes de pasar de verde a rojo. La operación "ponerLuces" admite dos parámetros, que son el color que tomar cada una de las dos hileras en un cambio de color. Los semáforos N y S están coordinados de 1 que siempre tienen las mismas luces encendidas; igual ocurre con los semáforos E y W. La Figura 7.4 muestra el flujo de vehículos girando a la izquierda en la calle N-S.

Cada detector vigila el carril de giro a la izquierda, informando al controlador de que hay un vehiculo detenido esperando a que el semáforo le permita el giro. La operación "detectarVehículo" del centro sirve para que un detector le informe de que efectivamente hay un vehículo esperando en la posición(N,S,E, W) especificada en el parámetro. El controlador utiliza la información de los cuatro detectores para optimizar el funcionamiento del cruce, de modo que si no hay coches esperando para girar a la izquierda se prescinde de la parte del ciclo correspondiente.

Se pide construir el diagrama de secuencia y el de colaboración asociado al control de los semáforo el cruce que refleje el paso de vehículos de Este a Oeste y viceversa y no lo permita en dirección Norte y Sur Norte. Así como el giro de vehículos en sentido Oeste Norte y Este Sur cuando se detecte un vehículo esperando para girar. Utilizando para ello los métodos que aparecen en el modelo de clases de la Figura y el diagrama que representa el funcionamiento del cruce de la Figura 7.4.

Solución

En la Figura 7.5 y Figura 7.6 se pueden ver los diagramas de secuencia y colaboración correspondiente dicho funcionamiento. En primer lugar hay que identificar el actor que inicia el proceso de control del cruce. En este ea trata de un actor interno al sistema que podemos denominar Reloj. A partir de que se pone en marcha i reloj, el control del cruce queda regulado por el controlador del cruce.

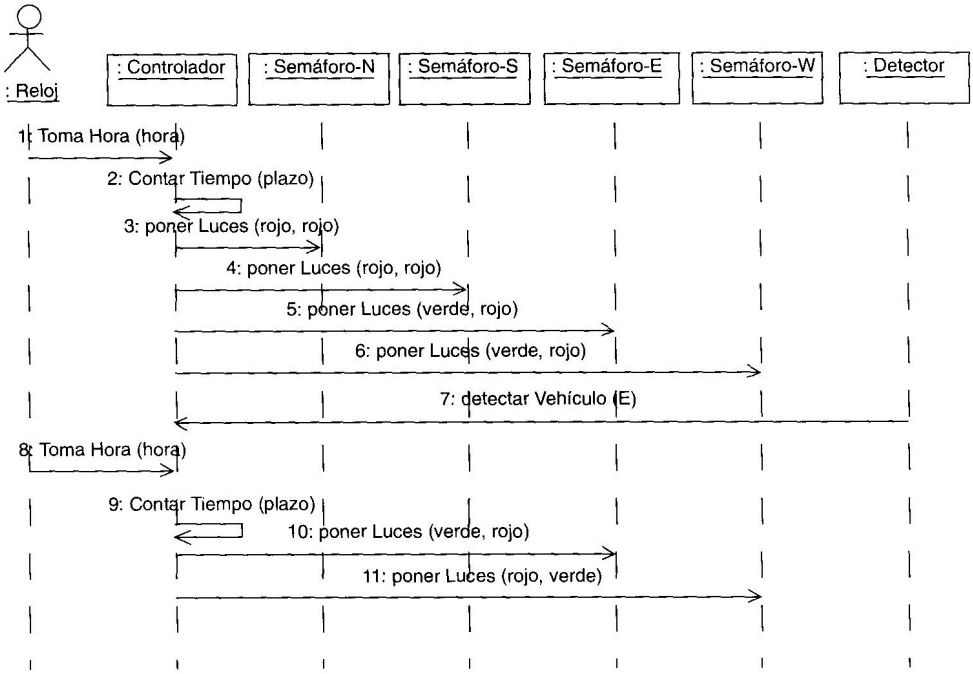


Figura 7.5: *Diagrama de Secuencia*

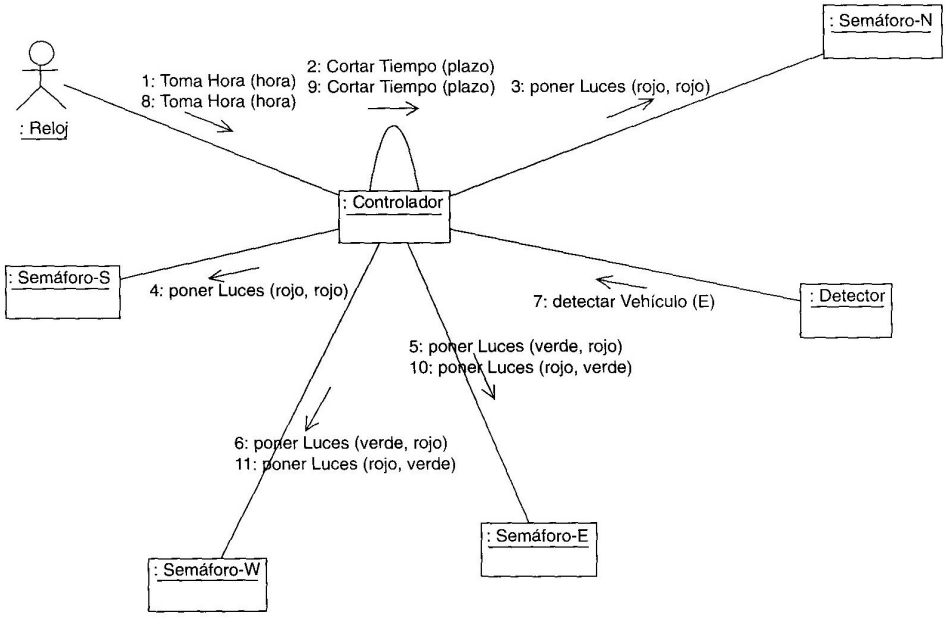


Figura 7.6: *Diagrama de Colaboración*

Tema 8. Diagrama de Estados. Ejercicios Resueltos

Ejercicio 1. Sistema de Solicitud de Dietas

Enunciado

El objetivo de este ejercicio es la identificación de los estados por los que pasarán tanto la clase Historia Clínica como la clase Módulo de Dietas (MAD). Dichos datos deben ser extraídos de la información que se adjunta sobre el funcionamiento de un sistema de atención médica que se pretende desarrollar.

Para poder llevar a cabo sus funciones, el MAD deberá poder consultar información sobre los pacientes (su historia clínica), las enfermedades y los posibles tratamientos (dietas). Para la obtención de las posibles dietas el MAD cuenta con un módulo subordinado (al que emite solicitudes) denominado DIETAS, que es el encargado de definir y prepocesar dietas para el MAD.

Un módulo no definido actualmente y denominado Gestor de Solicitudes (GS) es el encargado de solicitar un tratamiento al MAD, proporcionándole como única información el paciente a tratar.

El módulo de dietas (MAD) obtiene la historia clínica del paciente. La historia clínica del paciente sólo se facilita al MAD si dicho paciente está adscrito al servicio de Nutrición. En otro caso se produce una situación de excepción que se soluciona informando al MAD y éste a su vez al GS, dando de esta manera por finalizada la petición de tratamiento.

Para cada una de las enfermedades a tratar que recibe el módulo MAD emite una solicitud de dieta al módulo DIETAS, incluyendo en ella todos los datos necesarios para que se lleve a cabo con éxito.

El módulo DIETAS para cada una de las peticiones de dieta que recibe solicita información de todas las fuentes alimentarias asociadas a los nutrientes, cuyo déficit

produce la enfermedad a tratar. Que una vez recibe, le sirven para generar una dieta aconsejada, que envía al módulo de dietas (MAD).

Una vez que el módulo de dietas (MAD) recibe todas las dietas aconsejadas para todas las enfermedades para las cuales solicitó tratamiento. Las readapta teniendo en cuenta las condiciones características del caso que se está tratando y las une. Generando una dieta final, verificada, que es enviada al GS.

Solución

El diagrama de transición de estados correspondiente a la clase historia clínica es el que aparece en la Figura 8.1

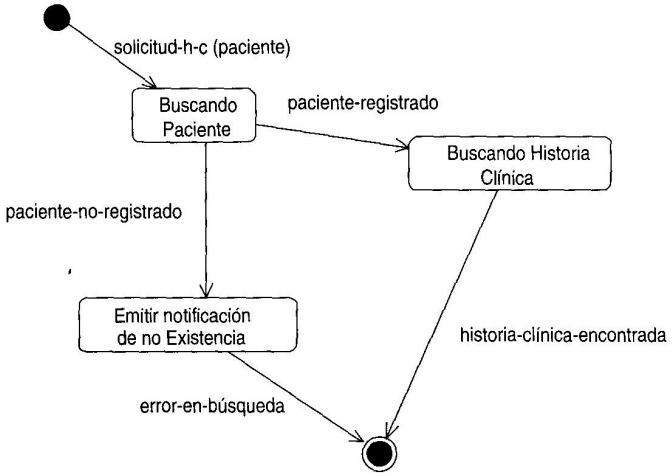


Figura 8.1: *Diagrama de transición de estados de la clase historia clínica*

Para realizar dicho diagrama el primer paso es la identificación del punto de comienzo en la vida de una historia clínica. Para ello se debe buscar en el enunciado el primer momento en que dicha clase es demandada para realizar alguna actividad. Ese momento marca el comienzo del diagrama de transición de estados y se produce cuando el MAD le solicita la historia clínica de un paciente determinado, a través del mensaje solicitud-h-c(paciente).

Desde ese momento hasta que la clase historia clínica comprueba la existencia de la historia clínica de ese paciente, la clase historia clínica está en el estado Buscando Paciente. En el caso en que el paciente no esté registrado, es decir, no se encuentre la historia clínica de dicho paciente, la clase historia clínica pasa al estado Emitir Notificación de no Existencia, tras el cual termina su actividad, o por decirlo de otro modo, pasa al estado de finalización.

Si por el contrario la historia clínica de dicho paciente existe, la clase historia clínica pasa al estado Buscar Historia Clínica, y se mantiene en dicho estado hasta que tenga todos los datos de la historia de dicho paciente. Una vez localizados todos los datos de dicha historia clínica, termina su actividad pasando al estado de fin.

Para el caso de la clase MAD, se identifica que el momento en que dicha clase es activada por primera vez es cuando el actor GS solicita un tratamiento a través del mensaje, solicitud-tratamiento (paciente), como se indica en la Figura 8.2.

En ese momento la clase MAD pasa al estado Solicitando-Historia-Clínica. Una vez que la clase MAD recibe el mensaje datos-historia-clínica se puede extraer del enunciado que pasa al estado solicitando-enfermedades. Se mantendrá en este estado hasta que alguna otra clase le envíe un mensaje indicándole cuáles son las enfermedades a tratar.

Una vez que recibe el mensaje enfermedades-a-tratar pasa al estado Solicitando Dieta. Hasta que la clase dietas no termine de confeccionar la dieta apropiada, la clase MAD permanecerá en el estado Solicitando Dietas.

Cuando la clase dietas le envíe a la clase MAD el mensaje dieta-final-revisada, la clase MAD pasará al estado Revisando Dieta, y cuando termine dicha revisión se la enviará al GS mediante el mensaje dieta-final revisada, tras cuya recepción pasará al estado final y su ciclo de vida terminará hasta la próxima solicitud de tratamiento de un paciente por parte del GS.

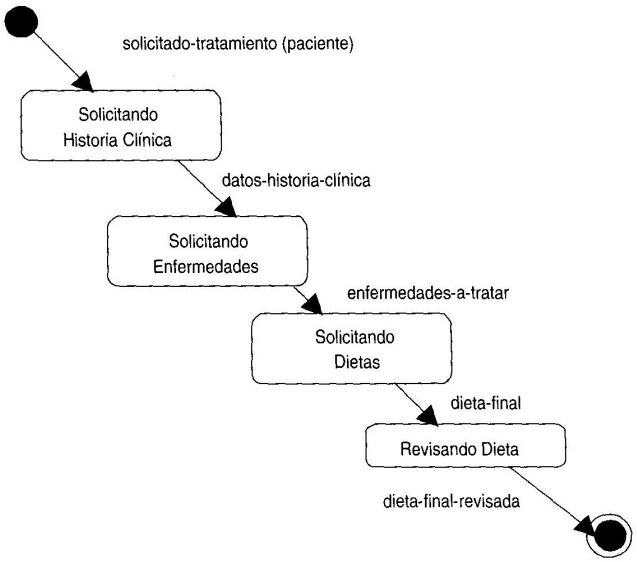


Figura 8.2: *Diagrama de transición de estado de la clase MAD*

Ejercicio 2. Gestión de un restaurante

Enunciado

Una cadena de restaurantes quiere automatizar el proceso de reservas así como el de los pedidos de cada mesa y la cantidad que hay en la cocina de cada uno de los

productos que se manejan para la realización de cada plato, y que obviamente han de ser repuestos desde el almacén a medida que éstos se van terminando.

Los clientes de los restaurantes pueden llamar por teléfono para reservar una mesa, pero lo que se está intentando poner de moda es el uso de unos terminales punto de reserva (TPR) ubicados en la calle. La ventaja que tiene el uso de estos terminales es la posibilidad de elegir la mesa en función de su ubicación dentro del restaurante, cosa que no se puede hacer por teléfono.

Las mesas están separadas en mesas de fumador, marcada con la F, y de no fumador, marcadas con NF. Además, cada mesa lleva un indicador con el número de personas para el que está pensada dicha mesa.

Si el cliente llega al restaurante veinte minutos después de la hora de reserva de la mesa, el sistema se encargará automáticamente de dejar libre dicha mesa.

Si no hay mesas libres a la hora indicada por el usuario, el TPR se lo comunica al cliente, dándole además la posibilidad de solicitar al sistema sugerencias sobre restaurantes disponibles a la hora y en el día solicitado.

Cuando un cliente llega a uno de los restaurantes de la cadena, se le pregunta si tiene reserva o no.

En el caso en que tenga reserva, bastará con que presente el ticket, si la hora de reserva no supera en veinte minutos a la hora de llegada al restaurante, la mesa pasa de estar libre a ocupada y se les sienta en el lugar que les corresponde.

Si por el contrario la hora de llegada supera en veinte minutos a la hora de reserva, el sistema se habrá encargado de anular dicha reserva de modo que la mesa haya quedado libre para otro posible cliente, por tanto, se les trata del mismo modo que si no tuvieran reserva. En ese caso el encargado en ese momento de las reservas solicita al sistema que le muestre las mesas libres para ese momento; si hay mesas libres, le pregunta al usuario si quiere mesa de fumador o de no fumador y cuántas personas son, el usuario se lo dice y en caso de que haya mesa libre, el encargado hace la reserva y les sienta. Si no hay mesa, el encargado le debe pedir al sistema el tiempo aproximado para que quede libre la próxima mesa de las características de la mesa solicitada. Esto podrá calcularlo el sistema a través del estado en que se encuentran las distintas mesas en un determinado momento; estos estados son:

Libre: si nadie la ha reservado.

Reservada: si alguien ha hecho una reserva.

Ocupada: si los comensales están ya a la mesa.

Pidiendo: sí el camarero está recogiendo el pedido de esa mesa.

En espera de comida: si están esperando que se les sirva.

Servidos: si los comensales ya tienen la comida en la mesa.

Esperando cuenta: si los comensales han pedido la cuenta.

Pagando: si los comensales ya tienen la cuenta en la mesa.

A partir de la información contenida en el enunciado se pide describir el comportamiento de la clase mesa, dentro del sistema de gestión de reservas.

Solución

El objetivo que persigue la cadena de restaurantes controlando el estado de sus mesas es poder dar mejor servicio a sus clientes no sólo a la hora de hacer una reserva por TPR o teléfono sino también cuándo; sin haber hecho una reserva previa, se acercan al restaurante para comer o cenar y en caso de que no haya una mesa libre con las condiciones que quieren los comensales, poder indicarles el tiempo que queda para que quede libre una mesa de las características indicadas.

Para poder llevar a cabo esta labor sin necesidad de que el encargado del restaurante tenga que ir mesa por mesa, que cumplan las características indicadas, viendo en ese momento lo que se encuentran haciendo las personas que están sentadas a la mesa, se quiere dotar al sistema de reservas de un módulo que permita hacer este proceso automáticamente. Para ello es necesario incluir en el sistema una clase MESA, cuyo ciclo de vida permita identificar, a partir del momento en que se encuentra actualmente, el tiempo que queda para que se quede libre. Para ello, es importante identificar el tiempo que los comensales tardan en realizar cada uno de los pasos que se llevan a cabo cuando están sentados a la mesa.

Dicho comportamiento se extrae del enunciado y quedará reflejado a través del diagrama de transición de estados de la clase MESA que aparece en la Figura 8.3.

El ciclo de vida de la clase mesa comienza cuando se abre el restaurante y se conecta el sistema de Reservas del restaurante. En ese momento la clase MESA pasa a estar en estado libre. En el momento en que los primeros comensales llegan al restaurante se les pregunta si tienen reserva o no.

En caso de que tengan la reserva se les sienta en la mesa correspondiente y, por lo tanto, esa mesa pasa a estar en el estado Pidiendo, en el cual estarán hasta que el camarero termine de tomarles nota. Cuando el camarero haya terminado de tomar nota la mesa pasa a un estado en el que los comensales están esperando a estar servidos, lo que implica que la clase MESA pase al estado Esperando Comida. Una vez que los platos están cocinados, el camarero les sirve y en ese momento el sistema pasa la clase MESA al estado Servidos, que durará mientras los comensales estén comiendo los platos que han pedido. Una vez que hayan terminado de comer y pidan la cuenta el sistema se encargará de poner esa MESA en estado de Esperando Cuenta, y permanecerá en este estado hasta que el camarero les traiga la cuenta y paguen, momento en el cual el sistema pasa dicha MESA al estado Libre. En este estado permanecerá mientras no lleguen nuevos comensales. En el

momento del cierre del restaurante, el ciclo de vida de esa MESA termina pasando al estado de fin.

Si los comensales que llegan no tienen mesa reservada y quieren comer o cenar, el encargado del restaurante les preguntará las características de la mesa que quieren e introducirá dichos datos en el sistema. El sistema irá comprobando por cada una de las mesas que cumplen con las características indicadas por los clientes y calculará el tiempo que queda para que queden libres en función del estado en que cada una de ellas se encuentra.

Del enunciado se puede extraer en conclusión que existe un estado más referente a la MESA cuando está Reservada, pero en realidad este estado no le corresponde a la clase MESA. El hecho de que una mesa esté o no reservada estará contemplado en otra clase, posiblemente la clase RESERVA, ya que una reserva es algo más que un simple estado, es la hora de la reserva, el nombre al que se hace la reserva, etc. Lo que sí existirá será una relación entre una mesa y todas sus reservas.

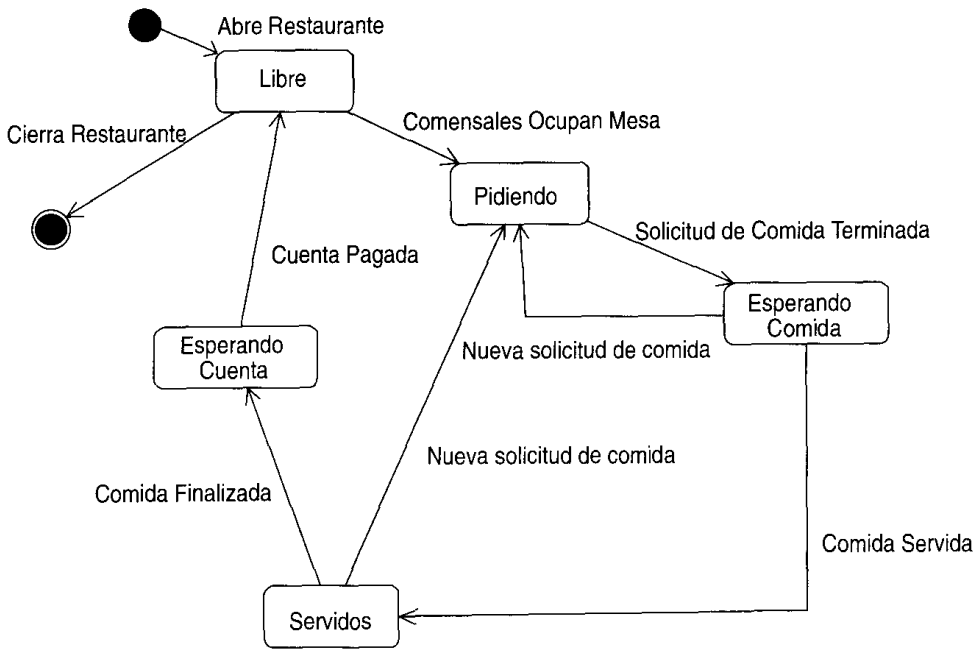


Figura 8.3: *Diagrama de transición de estado de la clase MESA.*

Ejercicio 4. Venta de Productos por Internet

Enunciado

El objetivo principal de este sistema es ofrecer la posibilidad de realizar fácil y eficazmente toda gama de acciones sobre marketing clásicas. Una empresa proveedora puede catalogar sus productos y ponerlos a disposición de sus clientes a través de la red y con la forma más real posible. El cliente puede

consultar el catálogo y efectuar varias operaciones a distancia sobre su contenido. Se pretende dar al cliente la posibilidad de visualizar estos productos en forma de tres dimensiones y dejarle toda la libertad de intervenir sobre el aspecto en tiempo real (color, dimensión, etc.). También se ofrece la posibilidad de hacer pedidos y seguir su evolución.

Los productos estarán en forma de objetos tridimensional (3D) y la consulta consistirá en visualizar y manipular esos objetos tridimensionales en tiempo real. El pedido se hace tomando en consideración las modificaciones que se han podido efectuar por el- cliente y almacenando dichas preferencias para que el personal de la empresa pueda consultarlas y procesar debidamente el pedido del cliente. Si, por ejemplo, lo que está comprando el cliente es un sofá y le ha puesto una tapicería de color azul, esas serán sus preferencias a la hora de hacer el pedido.

Cuando el usuario accede al sistema, el escenario o la escena en la que se visualizan los productos seleccionados está vacío, en el momento en que empieza a seleccionar productos éstos aparecen visualmente en la pantalla de su ordenador. El cliente tiene la posibilidad de interactuar con el sistema para cambiar el color de uno de los objetos visualizados, la textura o manipularlo. Las posibilidades que tiene de manipulación son: rotar el producto seleccionado, cambiarlo de dimensiones o cambiarlo de posición.

Lo que se pide en este ejercicio es construir el diagrama de transición de estados correspondiente al caso de uso que comienza cuando el cliente se conecta al sistema y selecciona algún producto para posteriormente interactuar con él en la escena en la que se visualizan los productos que va seleccionando.

Solución

La Figura 8.4 muestra un diagrama de estados para casos de uso y describe visualmente los estados del caso de uso Seleccionar y Manipular un producto. Esto permite garantizar el orden de los eventos externos del sistema.

El diagrama de transición de estados comienza cuando el cliente, ya conectado al sistema, está visualizando un escenario vacío debido a que no ha seleccionado aún ningún producto. En el momento en que añade el primer producto a la escena pasa al estado "Escena no vacía", en la que se estará visualizando el producto seleccionado.

Una vez que lo ha visualizado, puede realizar varias cosas:

? Si le gusta cómo está, lo envía directamente a la cesta de la compra para formar parte del pedido final. En este caso pasará al estado "Colocando Objeto en Cesta".

? Si lo que quiere es quitarlo de la escena porque ha decidido que no quiere visualizarlo, pasa al estado "Borrando Objeto 3D".

?Si lo que desea es realizar algún tipo de manipulación sobre un objeto de la escena, primeramente tendrá que seleccionar dicho objeto o producto de la escena. Para ello pasa al estado "Objeto 31) seleccionado", desde este estado tiene la posibilidad, según dice el enunciado, de:

?Cambiarlo de color o de textura, en cuyo caso pasará al estado "Cambiando color / textura".

?Cambiarlo de tamaño, posición o rotarlo, en cuyo caso pasará al estado "Objeto 31) manipulado", en el que tendrá la posibilidad de hacerle todos los cambios que desee al producto, hasta que éste se encuentre a su gusto, momento tras el cual podrá bien borrarlo de la escena porque finalmente no le agrade dicho producto o mandarlo a la cesta de la compra, para que forme parte del pedido final, manteniendo las características establecidas sobre el producto al haberlo manipulado.

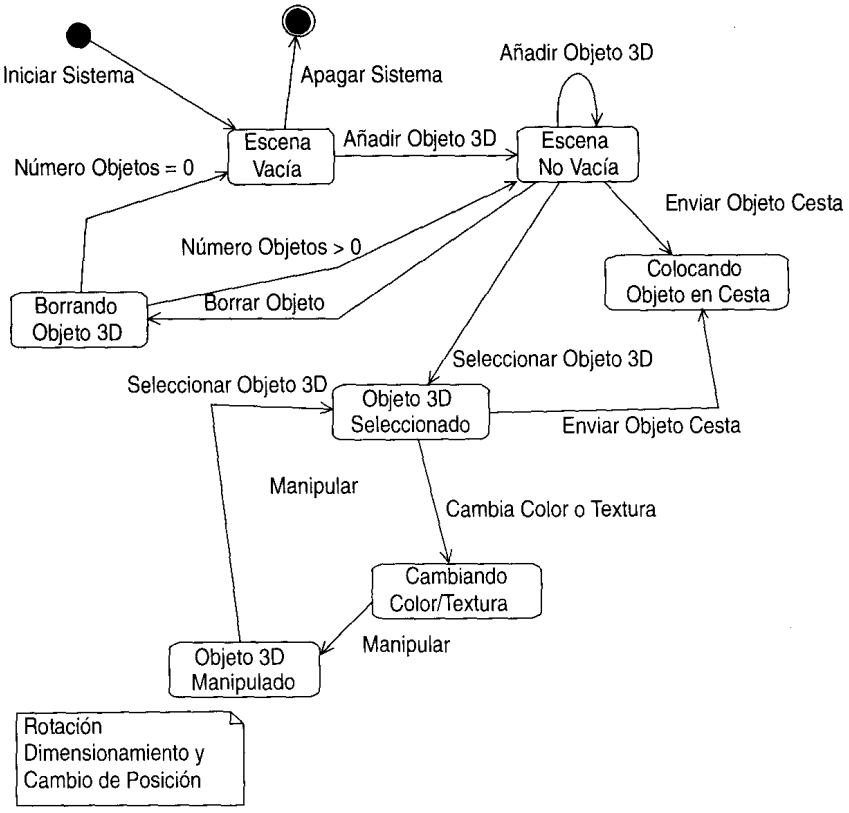


Figura 8.4: *Diagarama de Transición de Manipular y Seleccionar Producto*