



Este capítulo presenta brevemente los conceptos y los diagramas de UML, usando un ejemplo simple. El propósito del capítulo es organizar los conceptos de alto nivel de UML, en un pequeño conjunto de vistas y de diagramas que presentan los conceptos visualmente. Muestra cómo se utilizan los diferentes conceptos describir un sistema, y cómo encajan las vistas unas con otras. Este resumen no pretende ser completo; se omiten muchos conceptos. Para más detalles, véanse los capítulos siguientes, que desarrollan las vistas semánticas de UML, así como el material de referencia detallado en el capítulo de la enciclopedia.

El ejemplo es una taquilla de teatro que ha automatizado sus operaciones. Es un ejemplo inventado con el propósito es destacar varias construcciones de UML en un espacio breve. Está simplificado y deliberadamente no se presenta por completo. La presentación de un modelo completo, de un sistema implementado, ni cabría en un espacio tan pequeño ni destacaría una gama suficiente de construcciones, sin excesiva repetición.

Vistas de UML

No hay ninguna línea entre los diferentes conceptos y las construcciones en UML, pero, por conveniencia, nosotros los dividimos en varias vistas. Una vista es simplemente un subconjunto de UML que modela construcciones que representan un aspecto de un sistema. La división en diversas vistas es algo arbitraria, pero esperamos que sea intuitiva. Una o dos clases de diagramas proporcionan una notación visual para los conceptos de cada vista.

En el nivel superior, las vistas se pueden dividir en tres áreas: clasificación estructural, comportamiento dinámico, y gestión del modelo.

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

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

La gestión del modelo describe la organización de los propios modelos en unidades jerárquicas. El paquete es la unidad genérica de organización para los modelos. Los paquetes especiales incluyen a los modelos y a los subsistemas. La vista de gestión del modelo cruza las otras vistas y las organiza para el trabajo de desarrollo y el control de configuración.

Tabla 3.1 Vistas y diagramas de UML

Área	Vista	Diagramas	Conceptos Principales
estructural	vista estática	diagrama de clases	clase, asociación, generalización, dependencia, realización, interfaz
	vista de casos de uso	diagrama de casos de uso	caso de uso, actor, asociación, extensión, inclusión, generalización de casos de uso
	vista de implementación	diagrama de componentes	componente, interfaz, dependencia, realización
	vista de despliegue	diagrama de despliegue	nodo, componente, dependencia, localización
dinámica	vista de máquina de estados	diagrama de estados	estado, evento, transición, acción
	vista de actividad	diagrama de actividad	estado, actividad, transición de terminación, división, unión
	vista de interacción	diagrama de secuencia	interacción, objeto, mensaje, activación
		diagrama de colaboración	colaboración, interacción, rol de colaboración, mensaje
gestión del modelo	vista de gestión del modelo	diagrama de clases	paquete, subsistema, modelo
extensión de UML	todas	todos	restricción, estereotipo, valores etiquetados

UML también contiene varias construcciones previstas para proporcionar una capacidad limitada, pero útil, de extensión. Estas construcciones incluyen restricciones, estereotipos, y valores etiquetados y son aplicables a los elementos de todas las vistas.

La Tabla 3.1 muestra las vistas de UML y los diagramas que las muestran, así como los principales conceptos relevantes de cada vista. Esta tabla no se debe tomar como un rígido sistema de reglas sino simplemente como guía para el uso normal, dado que se permite la mezcla de vistas.

Vista estática

La vista estática modela los conceptos del dominio de la aplicación, así como los conceptos internos inventados como parte de la implementación de la aplicación. Esta visión es estática porque no describe el comportamiento del sistema dependiente del tiempo, que se describe en otras vistas. Los componentes principales de la vista estática son las clases y sus relaciones: asociación, generalización, y varias clases de dependencia, tales como realización y uso. Una clase es la descripción de un concepto del dominio de la aplicación o de la solución de la apli-

cación. Las clases son el centro alrededor del cual se organiza la vista de clases; otros elementos pertenecen o se unen a las clases. La visión estática se exhibe en los diagramas de clases, llamados así porque su objetivo principal es la descripción de clases.

Las clases se dibujan como rectángulos. Las listas de atributos y de operaciones se muestran en compartimentos separados. Los compartimentos pueden ser suprimidos cuando no es necesario el detalle completo. Una clase puede aparecer en varios diagramas. Sus atributos y operaciones se suprime, a menudo en todos menos en un diagrama.

Las relaciones entre clases se dibujan como las líneas que conectan rectángulos de clases. Las diversas clases de relaciones se diferencian por la textura de la línea y por los adornos en las líneas o en sus extremos.

La Figura 3.1 muestra un diagrama de clases de la aplicación de la taquilla. Este diagrama contiene parte del modelo del dominio “venta de entradas”. Muestra varias clases importantes, tales como **Cliente**, **Reserva**, **Entrada**, y **Representación**. Los clientes pueden tener muchas reservas, pero cada reserva es hecha por un cliente. Las reservas son de dos clases: suscripción y reservas individuales. Ambos reservan entradas: en un caso, solamente una entrada; en el otro caso, varias entradas. Cada entrada es parte de una suscripción o de una reserva individual, pero

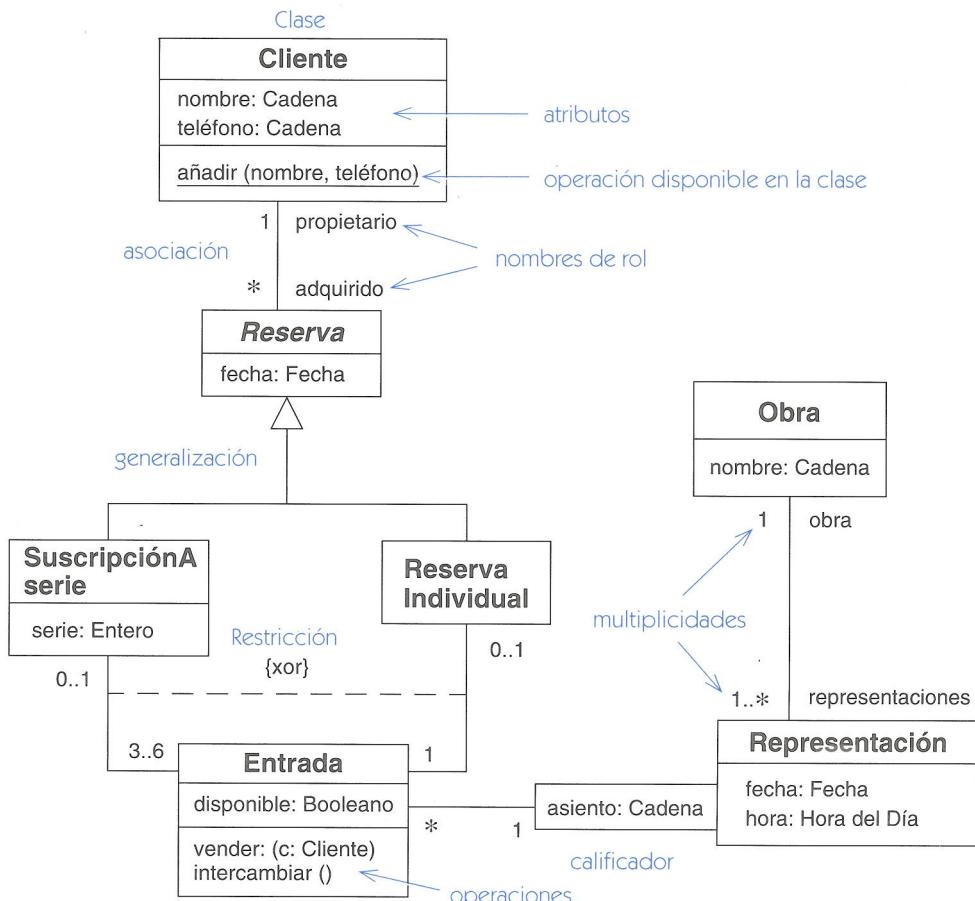


Figura 3.1 Diagrama de clases

no de ambas. Cada representación tiene muchas entradas disponibles, cada una con un número de asiento único. Una representación se puede identificar por una obra, una fecha, y una hora.

Las clases se pueden describir con varios niveles de precisión y concreción. Al empezar el diseño, el modelo captura los aspectos más lógicos del problema. En las fases posteriores, el modelo también capta decisiones de diseño y detalles de la implementación. La mayoría de las vistas tienen un comportamiento evolutivo similar.

Vista de los casos de uso

La vista de los casos de uso modela la funcionalidad del sistema según lo perciben los usuarios externos, llamados actores. Un caso de uso es una unidad coherente de funcionalidad, expresada como transacción entre los actores y el sistema. El propósito de la vista de casos de uso es enumerar a los actores y los casos de uso, y demostrar qué actores participan en cada caso de uso.

La Figura 3.2 muestra un diagrama de casos de uso para el ejemplo de la taquilla. Los actores son el vendedor, el supervisor, y el quiosco. El quiosco es otro sistema que acepta pedidos de un cliente. El cliente no es actor en la aplicación de la taquilla, porque no está conectado directamente con la aplicación. Los casos de uso incluyen el comprar entradas, a través del quiosco o del vendedor, comprar suscripciones (solamente a través del vendedor), y examinar las ventas totales (a petición del supervisor).

El comprar entradas y el comprar las suscripciones incluyen un fragmento común —que es hacer cargos al servicio de la tarjeta de crédito—. (La descripción completa de la taquilla im-

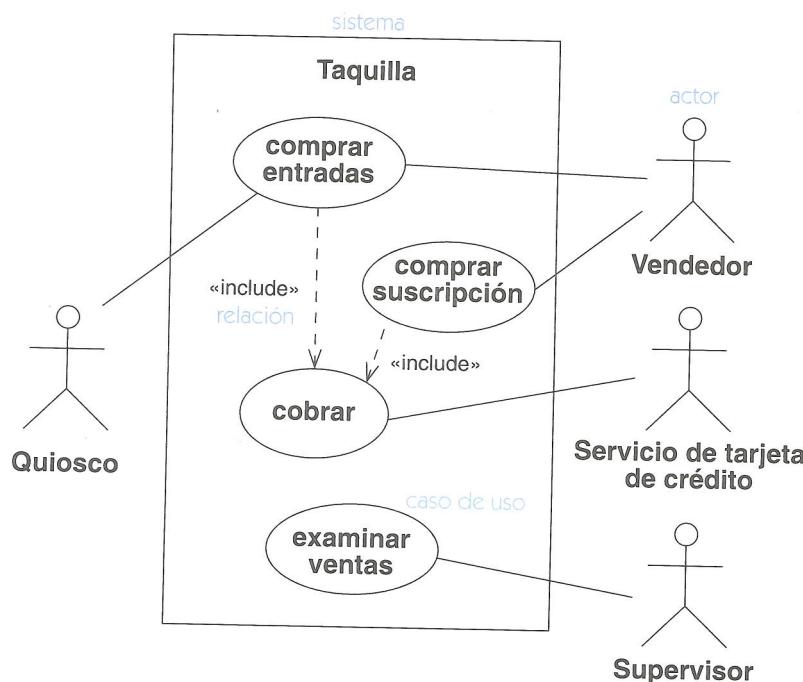


Figura 3.2 Diagrama de casos de uso

plicaría muchos otros casos de uso, tales como cambio de entradas y comprobación de disponibilidad.)

Los casos de uso se pueden también describir en varios niveles de detalle. Se pueden sacar partes como factor común y ser descritos en términos de otros casos de uso más simples. Un caso del uso se implementa como una colaboración en la vista de interacción.

Vista de interacción

La vista de interacción describe secuencias de intercambios de mensajes entre los roles que implementan el comportamiento de un sistema. Un rol de clasificador, o simplemente “rol”, es la descripción de un objeto, que desempeña un determinado papel dentro de una interacción, distinto de los otros objetos de la misma clase. Esta visión proporciona una vista integral del comportamiento de un sistema —es decir, muestra el flujo de control a través de muchos objetos—. La vista de interacción se exhibe en dos diagramas centrados en distintos aspectos: diagramas de secuencia y diagramas de colaboración.

El diagrama de secuencia

Un diagrama de secuencia muestra un conjunto de mensajes, dispuestos en una secuencia temporal. Cada rol en la secuencia se muestra como una línea de vida, es decir, una línea vertical

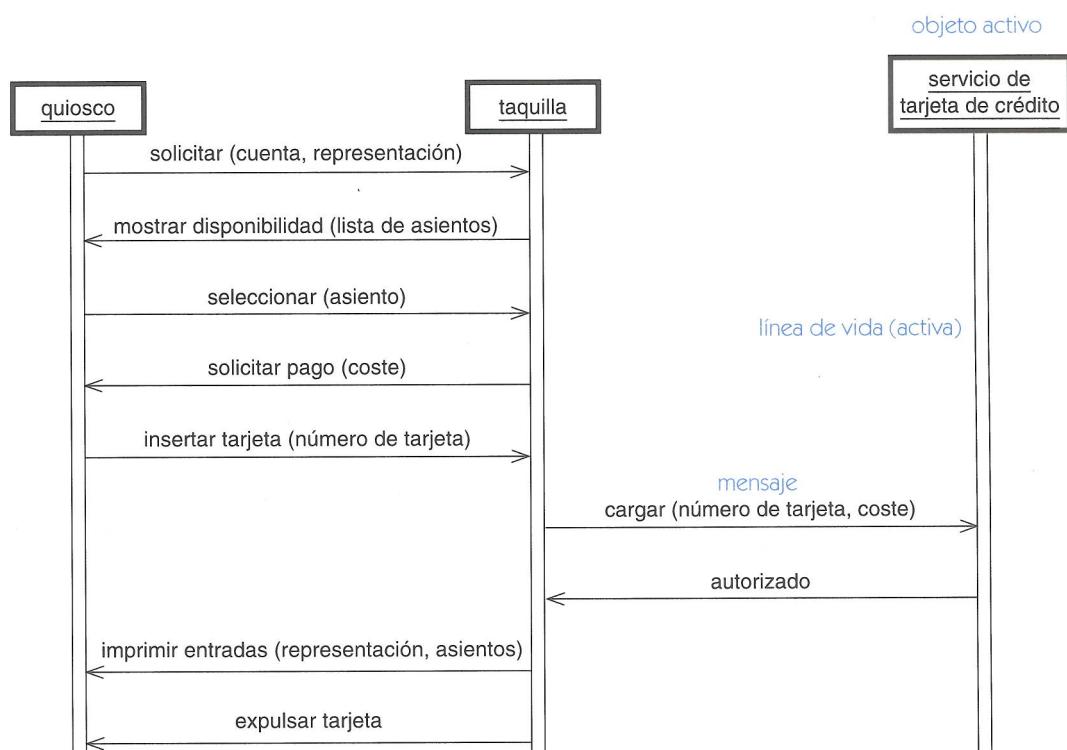


Figura 3.3 Diagrama de secuencia

que representa el rol durante cierto plazo de tiempo, con la interacción completa. Los mensajes se muestran como flechas entre las líneas de vida. Un diagrama de secuencia puede mostrar un escenario, es decir, una historia individual de una transacción.

Un uso de un diagrama de secuencia es mostrar la secuencia del comportamiento de un caso del uso. Cuando está implementado el comportamiento, cada mensaje en un diagrama de secuencia corresponde a una operación en una clase, a un evento disparador, o a una transición en una máquina de estados.

La Figura 3.3 muestra un diagrama de secuencia para el caso de uso **comprar entrada**. Este caso de uso lo inicia el cliente en el quiosco, comunicándose con la taquilla. Los pasos para el caso de uso **hacer cargos** se incluyen en la secuencia, que implica la comunicación con el quiosco y el servicio de la tarjeta de crédito. Este diagrama de secuencia está en una primera etapa de desarrollo y no muestra los detalles completos de la interfaz de usuario. Por ejemplo, la forma exacta de la lista de asientos y el mecanismo de especificar asientos, deben ser definidos aún, pero la comunicación esencial de la interacción ha sido especificada por el caso de uso.

El diagrama de colaboración

Una colaboración modela los objetos y los enlaces significativos dentro de una interacción. Los objetos y los enlaces son significativos solamente en el contexto proporcionado por la interacción. Un rol describe un objeto, y un rol en la asociación describe un enlace dentro de una colaboración. Un diagrama de colaboración muestra los roles en la interacción en una disposición geométrica (Figura 3.4). Los mensajes se muestran como flechas, ligadas a las líneas de la relación, que conectan a los roles. La secuencia de mensajes, se indica con los números secuenciales que preceden a las descripciones del mensaje.

Un uso de un diagrama de colaboración es mostrar la implementación de una operación. La colaboración muestra los parámetros y las variables locales de la operación, así como asociaciones más permanentes. Cuando se implementa el comportamiento, la secuencia de los mensajes corresponde a la estructura de llamadas anidadas y el paso de señales del programa.

La Figura 3.4 muestra un diagrama de colaboración para la interacción de la reserva de entradas en una fase ulterior del desarrollo. La colaboración muestra la interacción entre objetos internos en la aplicación para reservar entradas. La petición llega del quiosco y se utiliza para encontrar la base de datos de la representación deseada en el conjunto de todas las representaciones. El puntero **db** que es devuelto al objeto **vendedor de entradas** representa un enlace local transitorio a una base de datos de representaciones, que persiste durante la interacción y después se desecha. El vendedor de entradas solicita un número de asientos para la representación; se encuentra una selección de asientos en varias gamas de precios, se bloquea temporalmente, y se devuelve al quiosco para la selección de los clientes. Cuando el cliente hace una selección de la lista de asientos, se obtienen los asientos seleccionados y el resto son liberados.

Tanto los diagramas de secuencia como los diagramas de colaboración muestran interacciones, pero acentúan aspectos diferentes. Un diagrama de secuencia muestra secuencias en el tiempo como dimensión geométrica, pero las relaciones entre roles son implícitas. Un diagrama de colaboración muestra las relaciones entre roles geométricamente, y relaciona los mensajes con las relaciones, pero las secuencias temporales están menos claras, porque vienen dadas por los números de secuencia. Cada diagrama debe ser utilizado cuando su aspecto principal es el foco de atención.

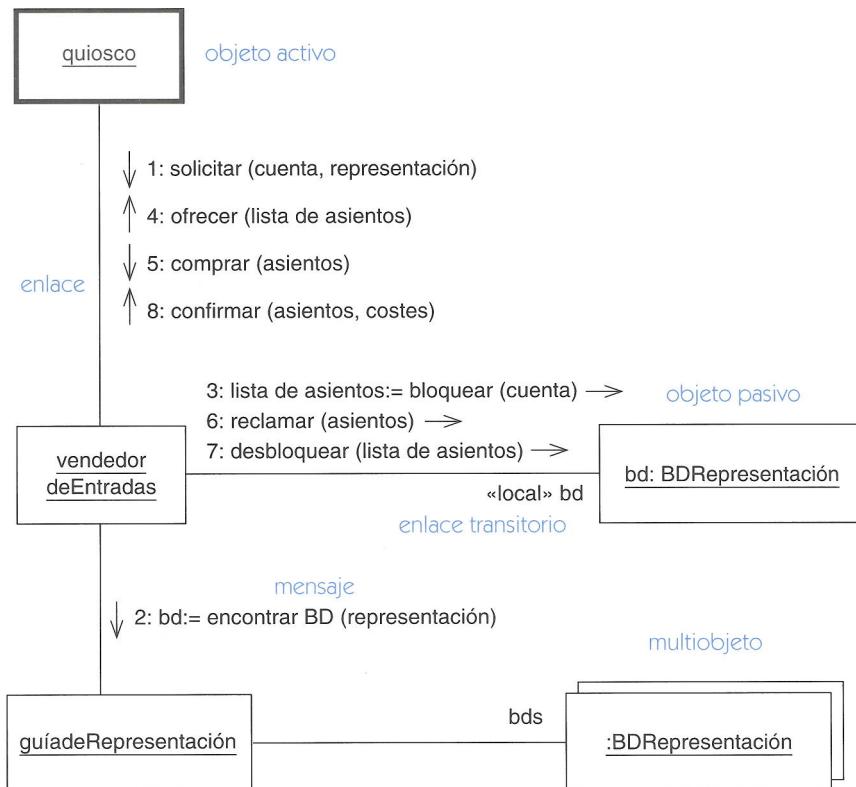


Figura 3.4 Diagrama de colaboración

Vista de la máquina de estados

Una máquina de estados modela las posibles historias de vida de un objeto de una clase. Una máquina de estados contiene los estados conectados por transiciones. Cada estado modela un período de tiempo, durante la vida de un objeto, en el que satisface ciertas condiciones. Cuando ocurre un evento, se puede desencadenar una transición que lleve el objeto a un nuevo estado. Cuando se dispara una transición, se puede ejecutar una acción unida a la transición. Las máquinas de estados se muestran como diagramas de estados.

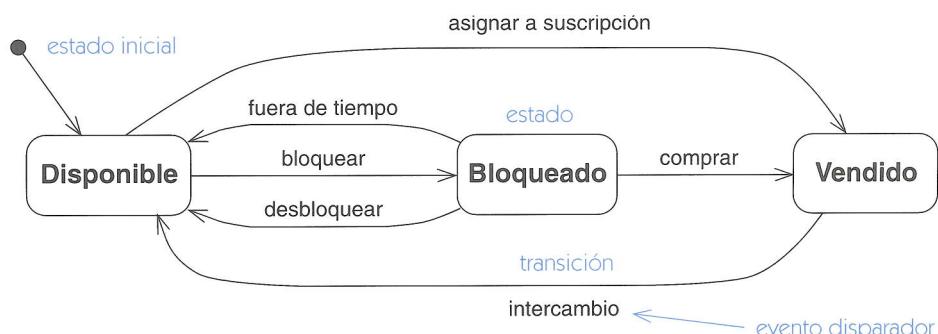


Figura 3.5 Diagrama de estados

La Figura 3.5 muestra un diagrama de estados de la historia de una entrada para una representación. El estado inicial de una entrada (representado por un punto negro) es el estado **disponible**. Antes del comienzo de la temporada, se asignan los asientos para los suscriptores de la temporada. Las entradas individuales compradas interactivamente primero se bloquean mientras el cliente hace una selección. Después de esto, se venden o se liberan si no se eligen. Si el cliente tarda demasiado en hacer una selección, finaliza el tiempo de la transacción y se libera el asiento. Los asientos vendidos para suscriptores de temporada, se pueden cambiar para otras representaciones, en cuyo caso, vuelven a estar disponibles otra vez.

Las máquinas de estados se pueden utilizar para describir interfaces de usuario, controladores de dispositivo, y otros subsistemas reactivos. También pueden usarse para describir los objetos pasivos que pasan por varias fases cualitativas distintas, durante su tiempo de vida, cada una de las cuales tiene su propio comportamiento especial.

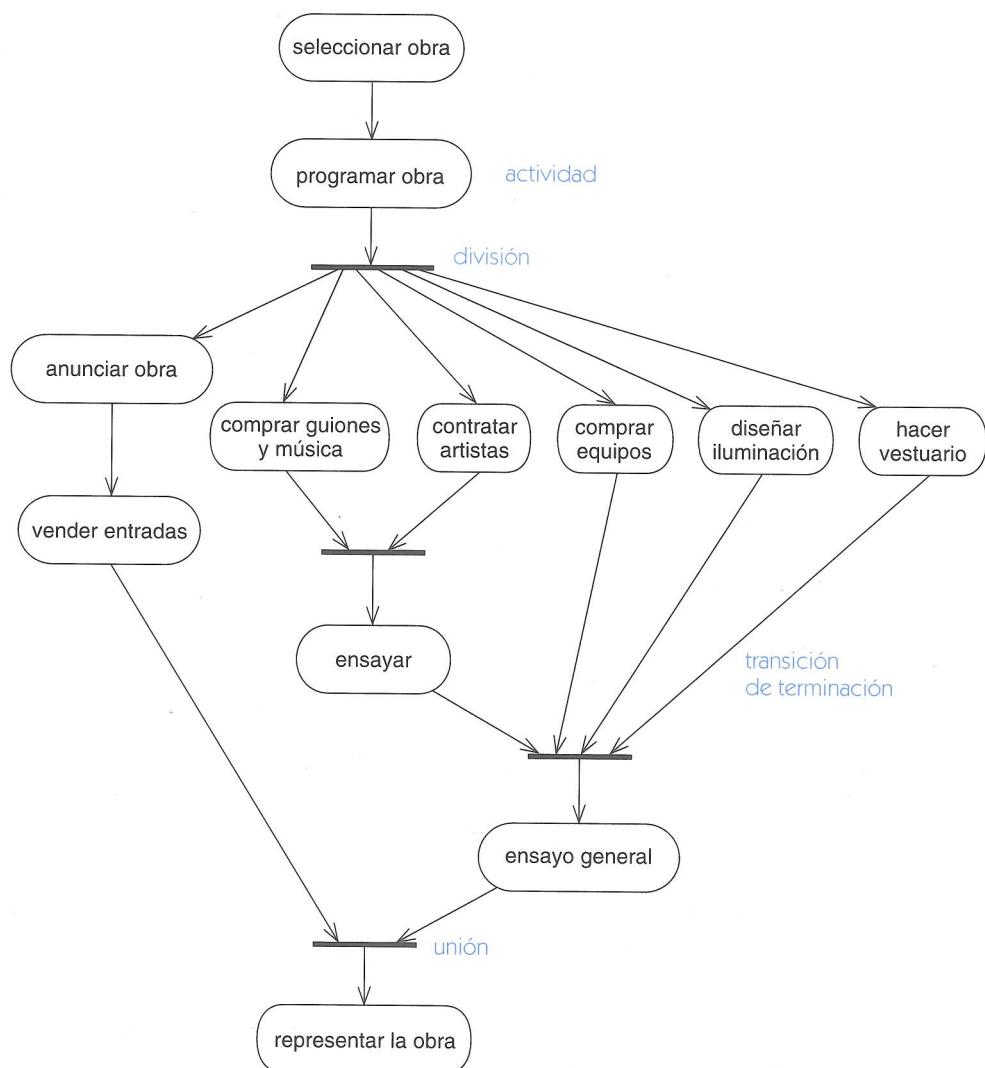


Figura 3.6 Diagrama de actividades

Vista de actividades

Un grafo de actividades es una variante de una máquina de estados, que muestra las actividades de computación implicadas en la ejecución de un cálculo. Un estado de actividad representa una actividad: un paso en el flujo de trabajo o la ejecución de una operación. Un grafo de actividades describe grupos secuenciales y concurrentes de actividades. Los grafo de actividades se muestran en diagramas de actividades.

La Figura 3.6 muestra un diagrama de actividades para la taquilla. Este diagrama muestra las actividades implicadas en montar una obra. (¡No tome este ejemplo demasiado en serio si tiene experiencia teatral!) Las flechas muestran dependencias secuenciales por ejemplo, las representaciones deben ser seleccionadas antes de poder planificarlas. Las barras horizontales representan bifurcaciones o uniones de control. Por ejemplo, después de planificar la obra, el teatro puede comenzar a darle publicidad, a comprar guiones, a contratar artistas, a construir decorados, a diseñar la iluminación, y a hacer los trajes, todo concurrentemente. Antes de que el ensayo pueda comenzar, sin embargo, se deben haber encargado los guiones y contratado a los artistas.

Este ejemplo muestra un diagrama de actividades, cuyo propósito es modelar los procesos reales de una organización humana. El modelado de tales negocios es un propósito importante de los diagramas de actividades, pero los diagramas de actividades se pueden también utilizar para modelar actividades software. Un diagrama de actividades es provechoso para entender el comportamiento de alto nivel de la ejecución de un sistema, sin profundizar en los detalles internos de los mensajes, lo que requeriría un diagrama de colaboración.

Los parámetros de entrada y de salida de una acción se pueden mostrar usando las relaciones de flujo que conectan la acción y un estado de flujo del objeto.

Vistas físicas

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

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

La vista de implementación modela los componentes de un sistema —a partir de los cuales se construye la aplicación— así como las dependencias entre los componentes, para poder determinar el impacto de un cambio propuesto. También modela la asignación de clases y de otros elementos del modelo a los componentes.

La vista de implementación se representa en diagramas componentes. La Figura 3.7 muestra un diagrama de componentes para el sistema de la taquilla.

Hay tres interfaces de usuario: la de los clientes que usan un quiosco, la de los vendedores que usan el sistema de reserva automatizado, y la de los supervisores que hacen consultas sobre las ventas de entradas. Hay un componente vendedor de entradas que ordena las peticiones de los quioscos y de los vendedores; un componente que procesa los cargos a la tarjeta de crédito; y la base de datos que contiene la información de la entrada. El diagrama de componentes mues-

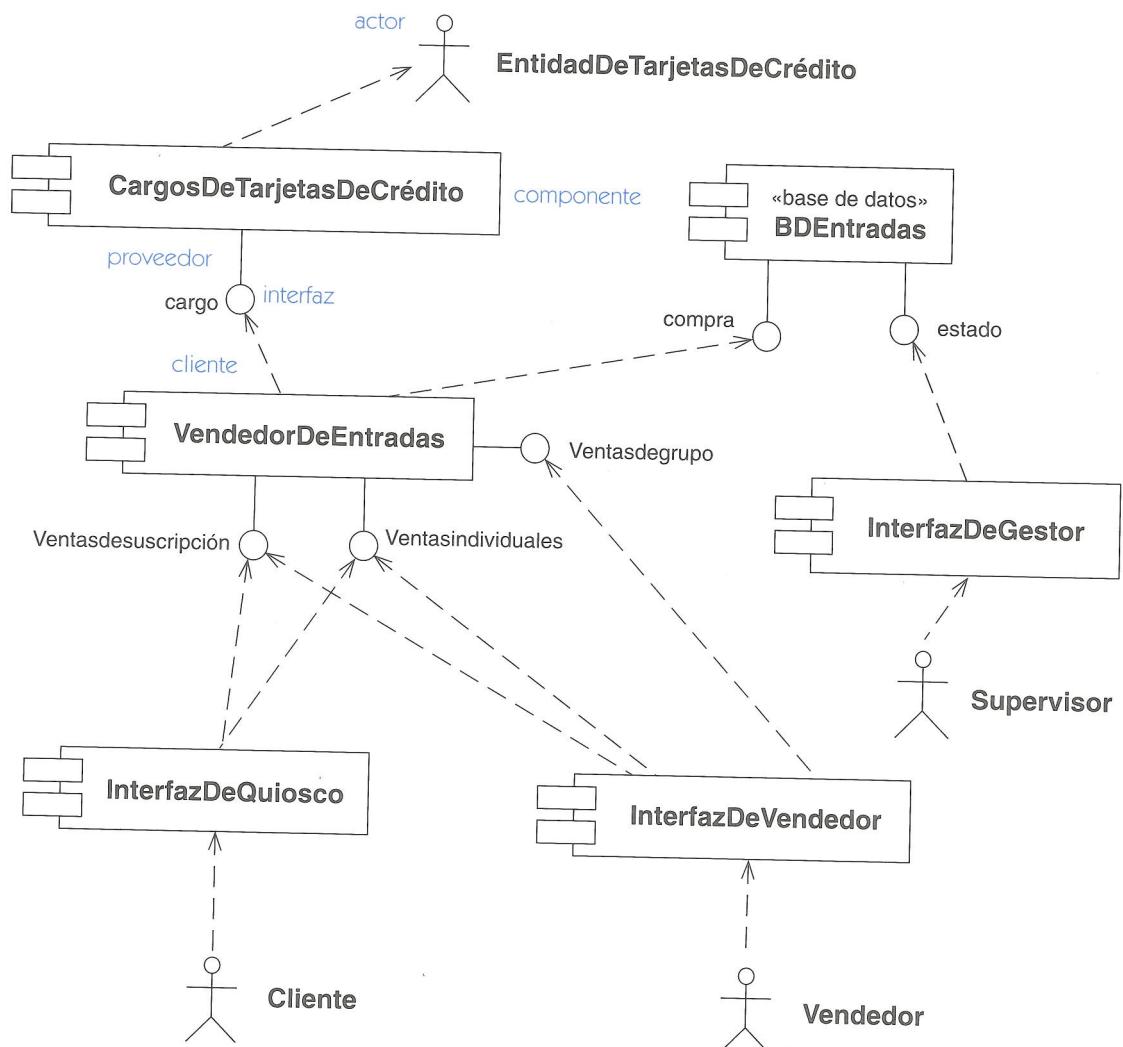


Figura 3.7 Diagrama de componentes

tra los tipos de componentes del sistema; una configuración particular de la aplicación puede tener más de una copia de un componente.

Un círculo pequeño con un nombre es una interfaz —un conjunto coherente de servicios—. Una línea sólida que va desde un componente a una interfaz, indica que el componente proporciona los servicios de la interfaz.

Una flecha de guiones de un componente a una interfaz indica que el componente requiere los servicios proporcionados por interfaz. Por ejemplo, las ventas de suscripciones y las ventas de grupos de entradas, son proporcionadas por el componente «vendedor de entradas»; las ventas de suscripciones son accesibles tanto para los quioscos como para los vendedores, pero las ventas de grupos sólo son accesibles para un vendedor.

La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos. Un nodo es un recurso de ejecución, tal como una computadora, un

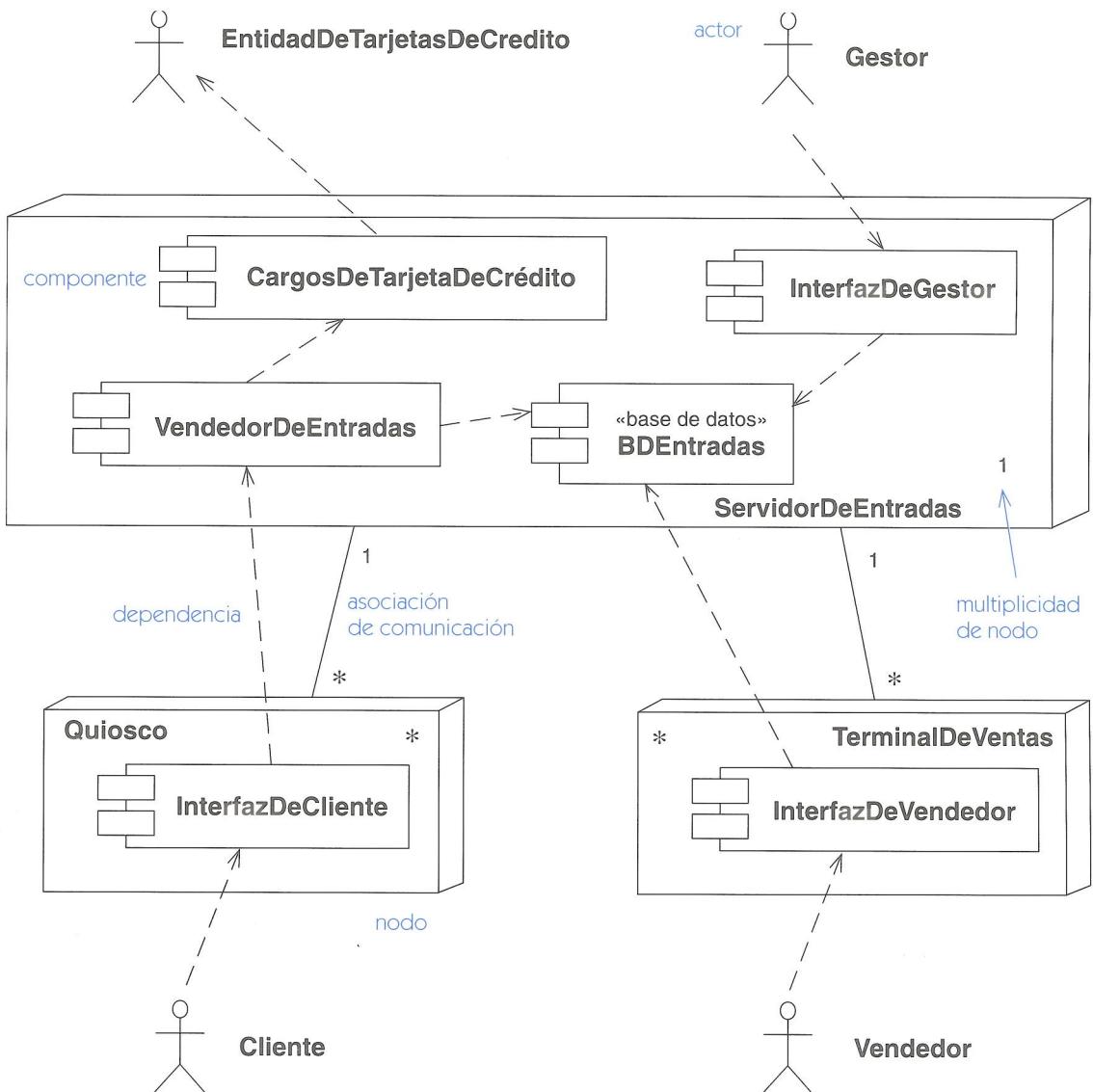


Figura 3.8 Diagrama de despliegue (nivel de descriptor)

dispositivo, o memoria. Esta vista permite determinar las consecuencias de la distribución y de la asignación de recursos.

La vista de despliegue se representa en diagramas de despliegue. La Figura 3.8 muestra un diagrama de despliegue del nivel de descriptor para el sistema de taquilla. Este diagrama muestra los tipos de nodos del sistema y los tipos de componentes que contienen. Un nodo se representa como un cubo.

La Figura 3.9 muestra un diagrama de despliegue del nivel de instancia, para el sistema de taquilla. El diagrama muestra los nodos individuales y sus enlaces, en una versión particular del sistema. La información de este modelo es consistente con la información del nivel de descriptor de la Figura 3.8.

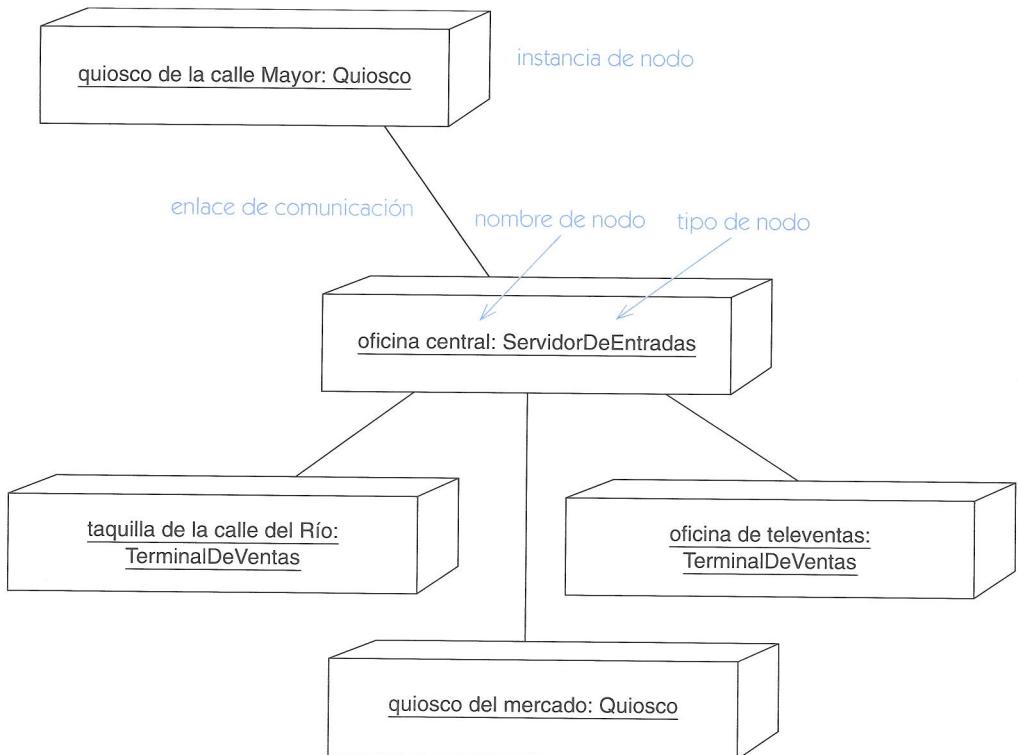


Figura 3.9 Diagrama de despliegue (nivel de instancia)

Vista de gestión del modelo

La vista de gestión del modelo modela la organización del modelo en sí mismo. Un modelo abarca un conjunto de paquetes que contienen los elementos del modelo, tales como clases, máquinas de estados, y casos de uso. Los paquetes pueden contener otros paquetes: por lo tanto, un modelo señala un paquete raíz, que contiene indirectamente todo el contenido del modelo. Los paquetes son unidades para manipular el contenido de un modelo, así como unidades para el control de acceso y el control de configuración. Cada elemento del modelo pertenece a un paquete o a otro elemento.

Un modelo es una descripción completa de un sistema, con una determinada precisión, desde un punto de vista. Puede haber varios modelos de un sistema desde distintos puntos de vista; por ejemplo, un modelo de análisis y un modelo de diseño. Un modelo se representa como una clase especial de paquete.

Un subsistema es otro paquete especial. Representa una porción de un sistema, con una interfaz perfectamente determinada, que puede ser implementado como un componente distinto.

Generalmente, la información de gestión del modelo se representa en diagramas de clases.

La Figura 3.10 muestra la descomposición de la totalidad del sistema de teatro en paquetes y sus relaciones de dependencia. El subsistema de taquilla incluye los ejemplos anteriores de este capítulo; el sistema completo también incluye operaciones del teatro y subsistemas de planificación. Cada subsistema consta de varios paquetes.

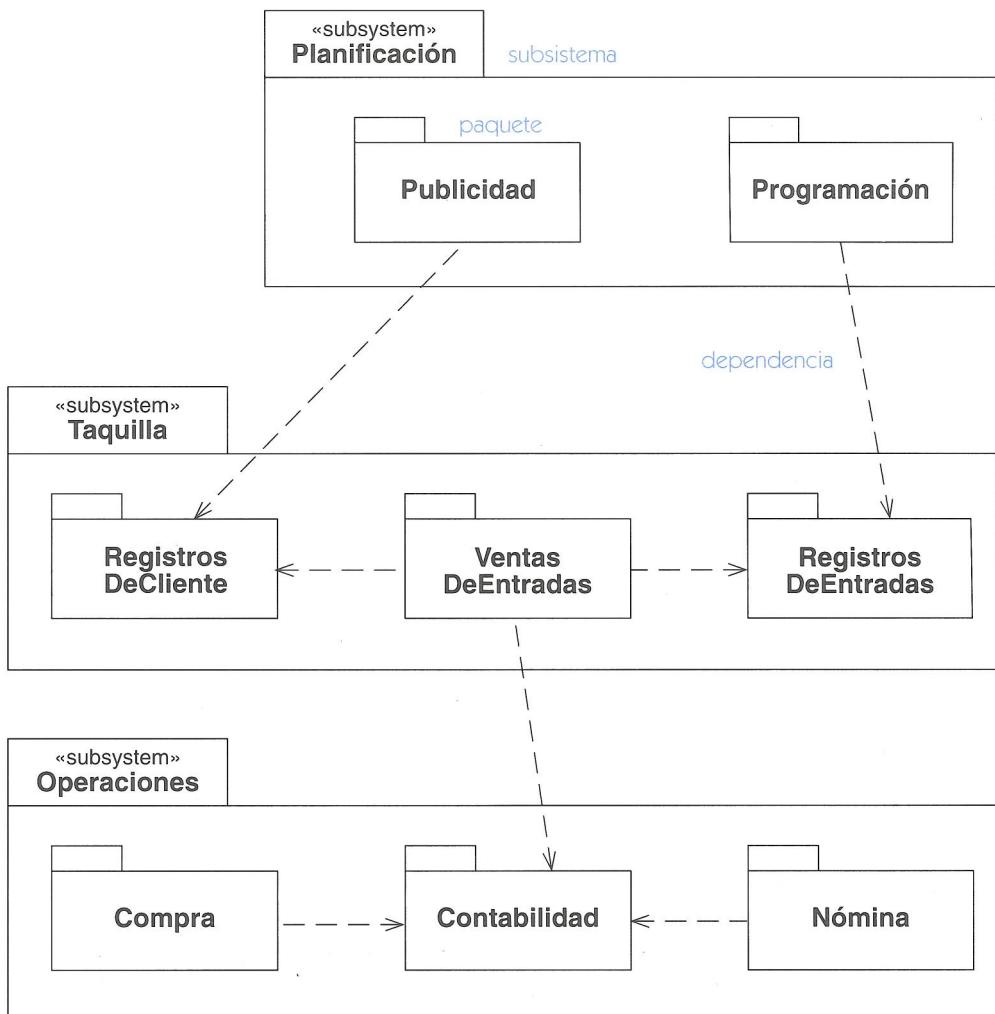


Figura 3.10 Paquetes

Construcciones de extensión

UML incluye tres construcciones principales de extensión: restricciones, estereotipos, y valores etiquetados. Una restricción es una declaración textual de una relación semántica expresada en un cierto lenguaje formal o en lenguaje natural. Un estereotipo es una nueva clase de elemento del modelo, ideada por el modelador, y basada en un tipo existente de elemento del modelo. Un valor etiquetado es una porción de información con nombre, unida a cualquier elemento del modelo.

Estas construcciones permiten muchas clases de extensiones a UML, sin requerir cambios al metamodelo básico de UML. Pueden ser utilizadas para crear versiones adaptadas a un área de aplicación.

La Figura 3.11 muestra ejemplos de restricciones, estereotipos, y de valores etiquetados. La restricción en la clase **Obra** asegura que los nombres de las obras sean únicos. La Figura 3.1

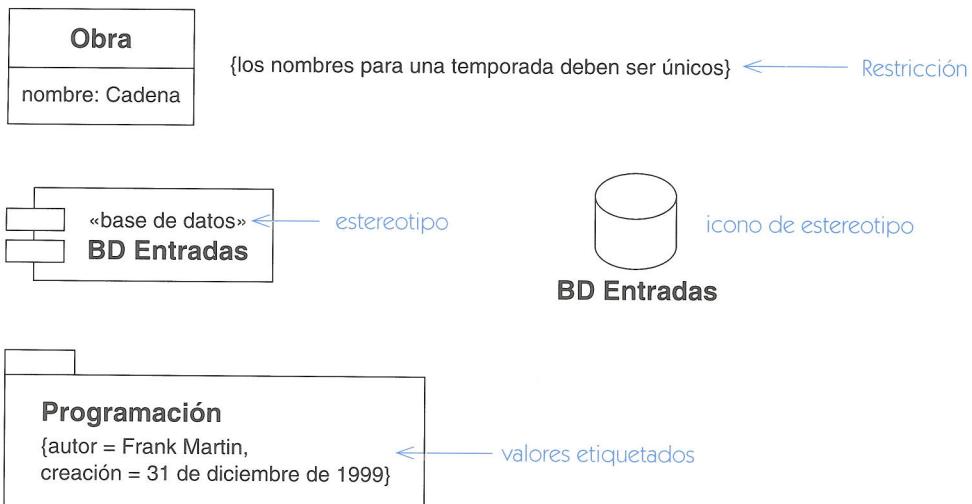


Figura 3.11 Construcciones de extensión

muestra una restricción **xor** entre dos asociaciones; un objeto sólo puede tener un enlace desde una de ellas a la vez. Las restricciones son útiles para representar sentencias que se pueden expresar en un lenguaje textual pero que no tienen soporte directo mediante construcciones de UML.

El estereotipo en el componente **DBEntrada** indica que el componente es una base de datos, lo que permite omitir qué interfaces soporta, puesto que son las soportadas por todas las bases de datos. Los modeladores pueden agregar nuevos estereotipos para representar elementos especiales. Un conjunto de restricciones implicadas, valores etiquetados, o características de la generación del código, pueden ser asociados a un estereotipo. Un modelador puede definir un ícono para un nombre de estereotipo, como ayuda visual, según lo mostrado en el diagrama. No obstante, la forma textual se puede utilizar siempre.

Los valores etiquetados en el paquete **Programación** muestran que Frank Martin es responsable de acabarla antes del final del milenio. Se puede unir a un elemento del modelo cualquier pieza de información arbitraria, como un valor etiquetado bajo un nombre elegido por el modelador. Los valores de texto son especialmente útiles para la información de gestión de proyecto y para los parámetros de generación del código. La mayoría de los valores etiquetados serían almacenados, como información emergente, dentro de una herramienta de edición, y no aparecerán generalmente en las figuras impresas.

Conexiones entre vistas

Dentro de un modelo coexisten distintas vistas y sus elementos tienen muchas conexiones, algunas de las cuales se muestran en la Tabla 3.2. Esta tabla no pretende ser completa, sino mostrar algunas de las relaciones principales entre elementos de diversas vistas.

Tabla 3.2 Algunas Relaciones entre Elementos de Diferentes Vistas

<i>Elemento</i>	<i>Elemento</i>	<i>Relación</i>
clase	máquina de estados	propiedad
operación	interacción	realización
caso de uso	colaboración	realización
caso de uso	instancia de interacción	escenario de ejemplo
instancia de un componente	instancia de un nodo	localización
acción	operación	llamada
acción	señal	envío
actividad	operación	llamada
mensaje	acción	invocación
paquete	clase	propiedad
rol	clase	clasificación