HORA 2

Orientación a objetos

Es fundamental que comprenda todo lo relacionado a la orientación a objetos para el proceso que realizará; específicamente, es importante que conozca algunos conceptos sobre la orientación a objetos.

En esta hora se tratarán los siguientes temas:

- Abstracción
- Herencia
- Polimorfismo
- Encapsulamiento o encapsulación
- Envío de mensajes
- Asociaciones
- Agregación

La orientación a objetos ha tomado por asalto y en forma legítima al mundo del software. Como medio para la generación de programas, tiene varias ventajas. Fomenta una metodología basada en componentes para el desarrollo

de software, de manera que primero se genera un sistema mediante un conjunto de objetos, luego podrá ampliar el sistema agregándole funcionalidad a los componentes que ya había generado o agregándole nuevos componentes, y finalmente podrá volver a utilizar los objetos que generó para el sistema cuando cree uno nuevo, con lo cual reducirá sustancialmente el tiempo de desarrollo de un sistema.

La orientación a objetos es tan importante para el diseño de software que el OMG (Grupo de administración de objetos), una corporación no lucrativa que establece las normas para el desarrollo orientado a objetos, predice que los ingresos obtenidos por el software orientado a objetos serán de 3 millardos de dólares en los siguientes tres a cinco años. El UML influye en esto al permitirle generar modelos de objetos fáciles de usar y comprender para que los desarrolladores puedan convertirlos en software.

La orientación a objetos es un paradigma (un paradigma que depende de ciertos principios fundamentales). En esta hora comprenderá dichos principios y verá qué es lo que hace funcionar a los objetos y cómo utilizarlos en el análisis y diseño. En la siguiente hora, empezará a aplicar el UML a tales principios.

Objetos, objetos por doquier

Los objetos concretos y virtuales, están a nuestro alrededor, ellos conforman nuestro mundo. Como indiqué en la hora anterior, el software actual simula al mundo (o un segmento de él), y los programas, por lo general, imitan a los objetos del mundo. Si comprende algunas cuestiones básicas de los objetos, entenderá cómo se deben mostrar éstos en las representaciones de software.

TERMINO NUEVO

Antes que nada, un objeto es la instancia de una clase (o categoría). Usted y yo, por ejemplo, somos instancias de la clase Persona. Un objeto cuenta con una *estructura*, es decir atributos (propiedades) y acciones. Las acciones son todas las *actividades* que el objeto es capaz de realizar. Los atributos y acciones, en conjunto, se conocen como *características* o *rasgos*.

Como objetos de la clase Persona, usted y yo contamos con los siguientes atributos: altura, peso y edad (puede imaginar muchos más). También realizamos las siguientes tareas: comer, dormir, leer, escribir, hablar, trabajar, etcétera. Si tuviéramos que crear un sistema que manejara información acerca de las personas (como una nómina o un sistema para el departamento de recursos humanos), sería muy probable que incorporáramos algunos de sus atributos y acciones en nuestro software.

En el mundo de la orientación a objetos, una clase tiene otro propósito además de la categorización. En realidad es una plantilla para fabricar objetos. Imagínelo como un molde de galletas que produce muchas galletas (algunos alegarían que esto es lo mismo que la categorización, pero evitemos dicho debate).

Regresemos al ejemplo de la lavadora. Si en la clase Lavadora se indica la marca, el modelo, el número de serie y la capacidad (junto con las acciones de agregar ropa, agregar detergente y sacar ropa), tendrá un mecanismo para fabricar nuevas instancias a partir de su clase; es decir, podrá crear nuevos objetos (vea la figura 2.1).

En la hora 3, "Uso de la orientación a objetos", verá que los nombres de las clases, como *lavadora*, se escribirán como Lavadora, y si constara de dos palabras se escribiría como LavadoraIndustrial, y las características como número *de serie* se escribirán como numeroSerie.

Esto es particularmente importante en el desarrollo de software orientado a objetos. Aunque este libro no se enfoca a la programación, le ayudará a comprender la orientación a objetos si sabe que las clases en los programas orientados a objetos pueden crear nuevas instancias.

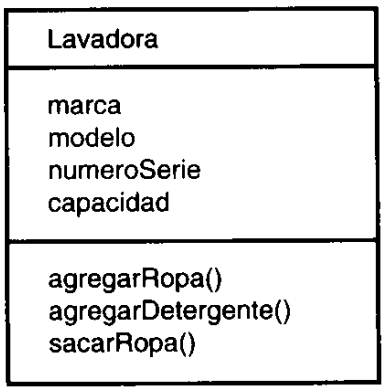
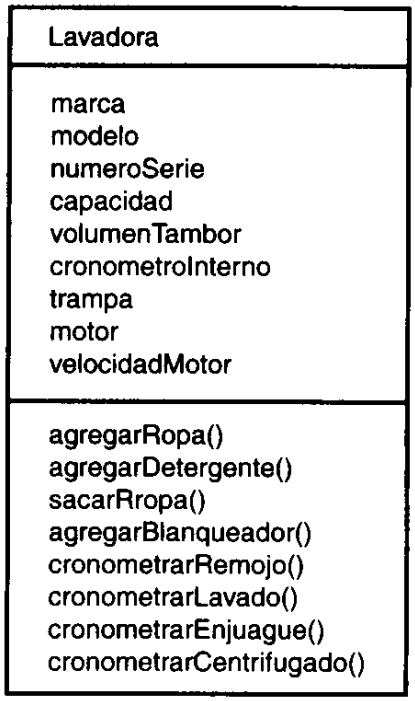


FIGURA 2.1
La clase Lavadora (modelo original) es una plantilla para generar nuevas instancias de Lavadoras.

Es importante que recuerde que el propósito de la orientación a objetos es desarrollar software que refleje particularmente (es decir, que modele) un esquema del mundo. Entre más atributos y acciones tome en cuenta, mayor será la similitud de su modelo con la realidad. En el ejemplo de la lavadora, tendrá un modelo más exacto si incluye los siguientes atributos: volumen del tambor, cronómetro interno, trampa, motor y velocidad del motor. Podría hacerlo más preciso si incluye las acciones de agregar blanqueador, cronometrar el remojo, cronometrar el lavado, cronometrar el enjuague y cronometrar el centrifugado (vea la figura 2.2).

FIGURA 2.2

La adición de atributos y acciones al modelo lo acerca a la realidad.



Algunos conceptos

La orientación a objetos se refiere a algo más que tan sólo atributos y acciones; también considera otros aspectos. Dichos aspectos se conocen como *abstracción*, *herencia*, *polimorfismo* y *encapsulamiento* o *encapsulación*. Otros aspectos importantes de la orientación a objetos son: el *envío de mensajes*, las *asociaciones* y la *agregación*. Examinemos cada uno de estos conceptos.

Abstracción

TERMINO NUEVO

La abstracción se refiere a quitar las propiedades y acciones de un objeto para dejar sólo aquellas que sean necesarias. ¿Qué significa esto último?

Diferentes tipos de problemas requieren distintas cantidades de información, aun si estos problemas pertenecen a un área en común. En la segunda fase de la creación de la clase Lavadora, se podrían agregar más atributos y acciones que en la primera fase. ¿Vale la pena?

Valdría la pena si usted pertenece al equipo de desarrollo que generará finalmente la aplicación que simule con exactitud lo que hace una lavadora. Un programa de este tipo (que podría ser muy útil para los ingenieros de diseño que actualmente estén trabajando en el diseño de una lavadora) deberá ser tan completo que permita obtener predicciones exactas respecto a lo que ocurriría cuando se fabrique la lavadora, funcione a toda su capacidad y lave la ropa. De hecho, para este caso podrá quitar el atributo del número de serie, dado que posiblemente no será de mucha ayuda.

Por otra parte, si va a generar un software que haga un seguimiento de las transacciones en una lavandería que cuente con diversas lavadoras, posiblemente no valdrá la pena. En este programa no necesitará todos los atributos detallados y operaciones del párrafo anterior, no obstante, quizá necesite incluir el número de serie de cada objeto Lavadora.

En cualquier caso, con lo que se quedará luego de tomar su decisión respecto a lo que incluirá o desechará, será una abstracción de una lavadora.

Herencia

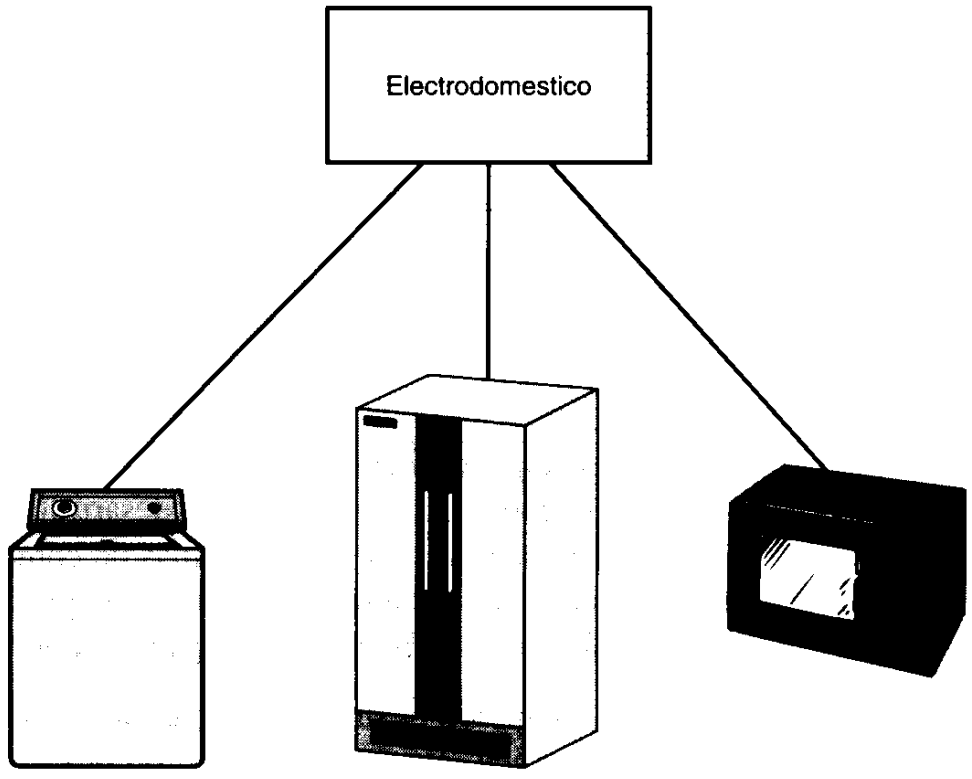
TERMINO NUEVO Como ya se mencionó anteriormente, una clase es una categoría de objetos (y en el mundo del software, una plantilla sirve para crear otros objetos). Un objeto es una instancia de una clase. Esta idea tiene una consecuencia importante: como instancia de una clase, un objeto tiene todas las características de la clase de la que proviene. A esto se le conoce como *herencia*. No importa qué atributos y acciones decida usar de la clase Lavadora, cada objeto de la clase heredará dichos atributos y operaciones.

Un objeto no sólo hereda de una clase, sino que una clase también puede heredar de otra. Las lavadoras, refrigeradores, hornos de microondas, tostadores, lavaplatos, radios, licuadoras y planchas son clases y forman parte de una clase más genérica llamada: Electrodomesticos. Un electrodoméstico cuenta con los atributos de interruptor y cable eléctrico, y las operaciones de encendido y apagado. Cada una de las clases Electrodomestico heredará los mismos atributos; por ello, si sabe que algo es un electrodoméstico, de inmediato sabrá que cuenta con los atributos y acciones de la clase Electrodomestico.

TERMINO NUEVO Otra forma de explicarlo es que la lavadora, refrigerador, horno de microondas y cosas por el estilo son *subclases* de la clase Electrodomestico. Podemos decir que la clase Electrodomestico es una *superclase* de todas las demás. La figura 2.3 le muestra la relación de superclase y subclase.

FIGURA 2.3

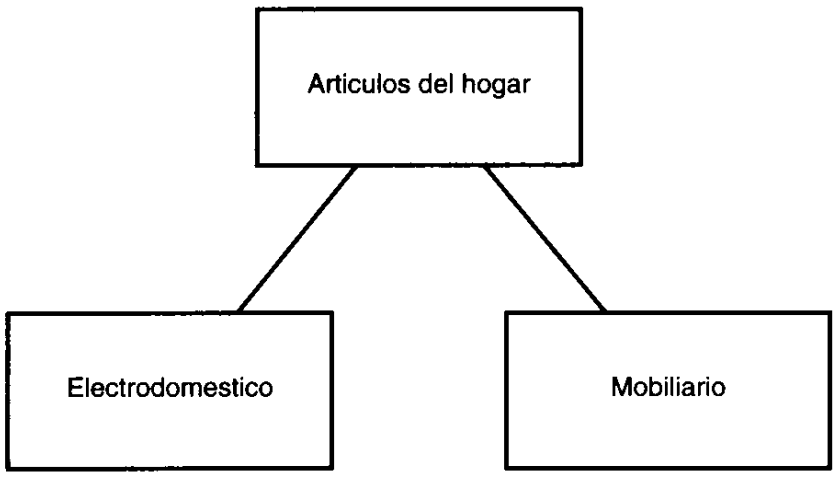
Los electrodomésticos heredan los atributos y acciones de la clase Electrodomestico. Cada electrodoméstico es una subclase de la clase Electrodomestico. La clase Electrodomestico es una superclase de cada subclase.



La herencia no tiene por qué terminar aquí. Por ejemplo, Electrodomestico es una subclase de Articulos del hogar, como le muestra la figura 2.4. Otra de las subclases de Articulos del hogar podría ser Mobiliario, que tendrá sus propias subclases.

FIGURA 2.4

Las superclases también pueden ser subclases, y heredar de otras superclases.



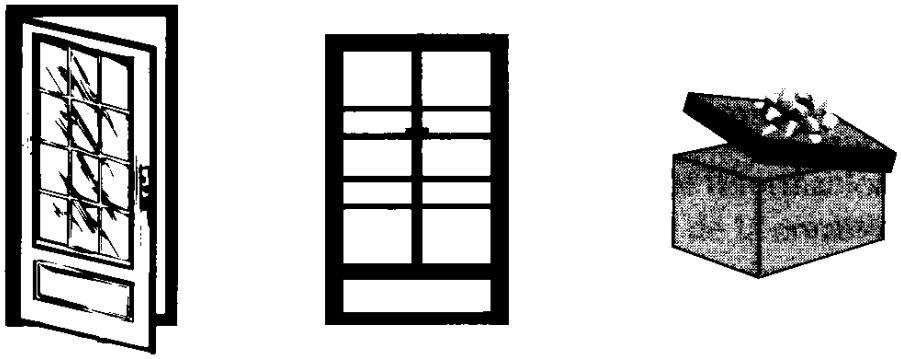
Polimorfismo

TERMINO NUEVO

En ocasiones una operación tiene el mismo nombre en diferentes clases. Por ejemplo, podrá abrir una puerta, una ventana, un periódico, un regalo o una cuenta de banco, en cada uno de estos casos, realizará una operación diferente. En la orientación a objetos, cada clase "sabe" cómo realizar tal operación. Esto es el *polimorfismo* (vea la figura 2.5).

FIGURA 2.5

En el polimorfismo, una operación puede tener el mismo nombre en diversas clases, y funcionar distinto en cada una.



En primera instancia, parecería que este concepto es más importante para los desarrolladores de software que para los modeladores, ya que los desarrolladores tienen que crear el software que implemente tales métodos en los programas de computación, y deben estar conscientes de diferencias importantes entre las operaciones que pudieran tener el mismo nombre. Y podrán generar clases de software que "sepan" lo que se supone que harán.

No obstante, el polimorfismo también es importante para los modeladores ya que les permite hablar con el cliente (quien está familiarizado con la sección del mundo que será modelada) en las propias palabras y terminología del cliente. En ocasiones, las palabras y terminología del cliente nos conducen a palabras de acción (como "abrir") que pueden

tener más de un significado. El polimorfismo permite al modelador mantener tal terminología sin tener que crear palabras artificiales para sustentar una unicidad innecesaria de los términos.

Encapsulamiento

En cierto comercial televisivo, dos personas discuten acerca de todo el dinero que ahorrarían si marcaran un prefijo telefónico en particular antes de hacer una llamada de larga distancia.

Uno de ellos pregunta con incredulidad: "¿Cómo funciona?"

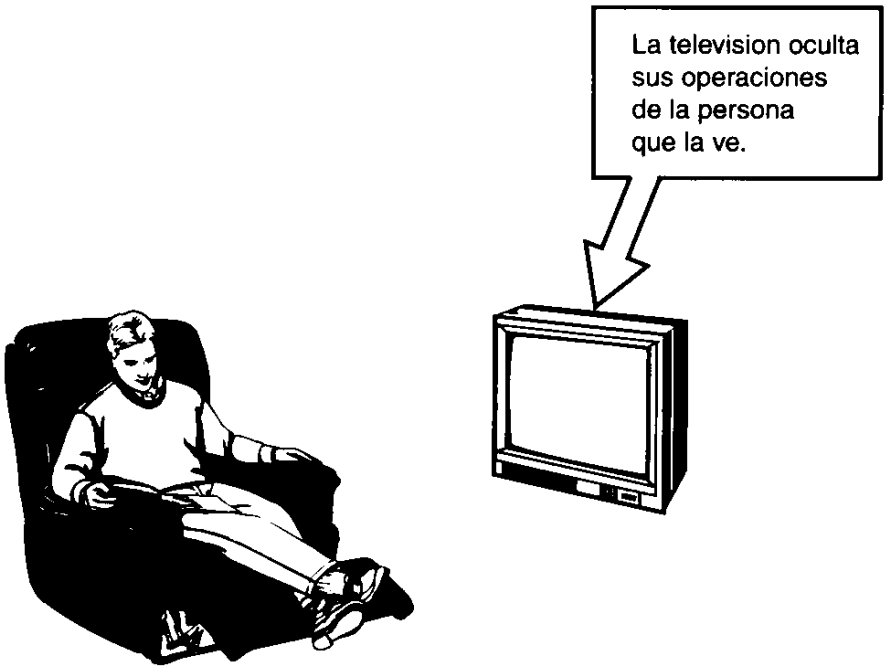
Y el otro responde: "¿Cómo hacen que las rosetas de maíz estallen? ¿A quién le importa?"

TÉRMINO NUEVO

La esencia del *encapsulamiento* (o encapsulación) es que cuando un objeto trae consigo su funcionalidad, esta última se oculta (vea la figura 2.6). Por lo general, la mayoría de la gente que ve la televisión no sabe o no se preocupa de la complejidad electrónica que hay detrás de la pantalla ni de todas las operaciones que tienen que ocurrir para mostrar una imagen en la pantalla. La televisión hace lo que tiene que hacer sin mostrarnos el proceso necesario para ello y, por suerte, la mayoría de los electrodomésticos funcionan así.

FIGURA 2.6

Los objetos encapsulan lo que hacen; es decir, ocultan la funcionalidad interna de sus operaciones, de otros objetos y del mundo exterior.



¿Cuál es la importancia de esto? En el mundo del software, el encapsulamiento permite reducir el potencial de errores que pudieran ocurrir. En un sistema que consta de objetos, éstos dependen unos de otros en diversas formas. Si uno de ellos falla y los especialistas de software tienen que modificarlo de alguna forma, el ocultar sus operaciones de otros objetos significará que tal vez no será necesario modificar los demás objetos.

En el mundo real, también verá la importancia del encapsulamiento en los objetos con los que trabaje. Por ejemplo, el monitor de su computadora, en cierto sentido, oculta sus operaciones de la CPU, es decir, si algo falla en su monitor, lo reparará o lo reemplazará; pero es muy probable que no tenga que reparar o reemplazar la CPU al mismo tiempo que el monitor.

TERMINO NUEVO

Ya que estamos en el tema, existe un concepto relacionado. Un objeto oculta lo que hace a otros objetos y al mundo exterior, por lo cual al encapsulamiento también se le conoce como *ocultamiento de la información*. Pero un objeto tiene que presentar un "rostro" al mundo exterior para poder iniciar sus operaciones. Por ejemplo, la televisión tiene diversos botones y perillas en sí misma o en el control remoto. Una lavadora tiene diversas perillas que le permiten establecer los niveles de temperatura y agua. Los botones y perillas de la televisión y de la lavadora se conocen como *interfaces*.

Envío de mensajes

Mencioné que en un sistema los objetos trabajan en conjunto. Esto se logra mediante el envío de mensajes entre ellos. Un objeto envía a otro un mensaje para realizar una operación, y el objeto receptor ejecutará la operación.

Una televisión y su control remoto pueden ser un ejemplo muy intuitivo del mundo que nos rodea. Cuando desea ver un programa de televisión, busca el control remoto, se sienta en su silla favorita y presiona el botón de encendido. ¿Qué ocurre? El control remoto le envía, literalmente, un mensaje al televisor para que se encienda. El televisor recibe el mensaje, lo identifica como una petición para encenderse y así lo hace. Cuando desea ver otro canal, presiona el botón correspondiente del control remoto, mismo que envía otro mensaje a la televisión (cambiar de canal). El control remoto también puede comunicar otros mensajes como ajustar el volumen, silenciar y activar los subtítulos.

Volvamos a las interfaces. Muchas de las cosas que hace mediante el control remoto, también las podrá hacer si se levanta de la silla, va a la televisión y presiona los botones correspondientes (¡alguna vez lo habrá hecho ya!). La interfaz que la televisión le presenta (el conjunto de botones y perillas) no es, obviamente, la misma que le muestra al control remoto (un receptor de rayos infrarrojos). La figura 2.7 le muestra esto.

Asociaciones

Otro acontecimiento común es que los objetos se relacionan entre sí de alguna forma. Por ejemplo, cuando enciende su televisor, en términos de orientación a objetos, usted se asocia con su televisor.

La asociación "encendido" es en una sola dirección (una vía), esto es, usted enciende la televisión, como se ve en la figura 2.8. No obstante, a menos que vea demasiada televisión, ella no le devolverá el favor. Hay otras asociaciones que son en dos direcciones, como "casamiento".

FIGURA 2.7

Ejemplo de un mensaje enviado de un objeto a otro: el objeto "control remoto" envía un mensaje al objeto "televisión" para encenderse. El objeto "televisión" recibe el mensaje mediante su interfaz, un receptor infrarrojo.

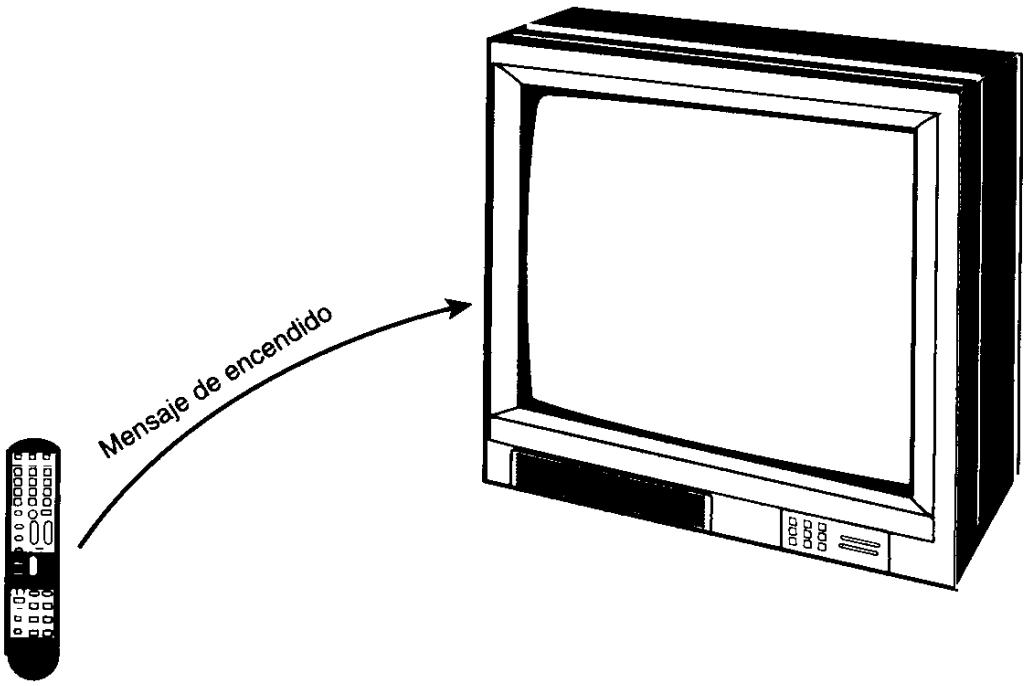
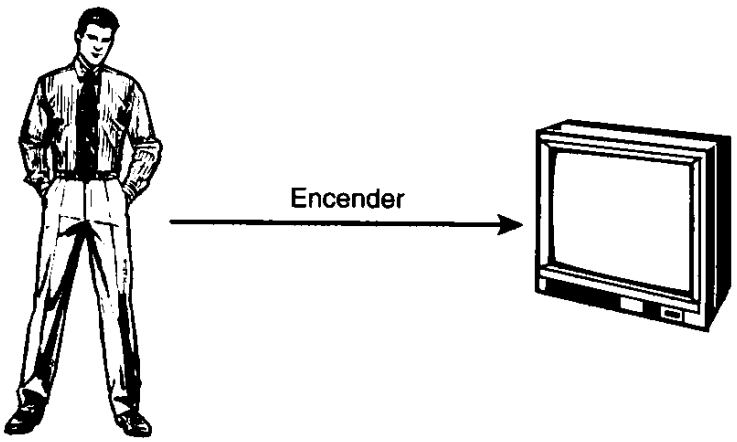


FIGURA 2.8

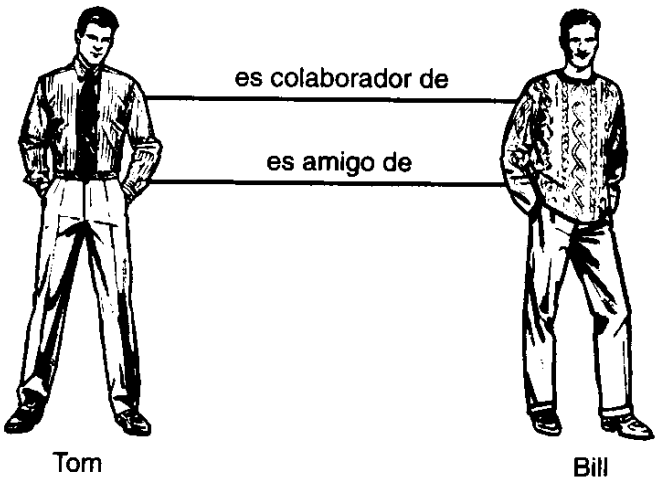
Con frecuencia los objetos se relacionan entre sí de alguna forma. Cuando usted enciende su televisión, tendrá una asociación en una sola dirección con ella.



En ocasiones, un objeto podría asociarse con otro en más de una forma. Si usted y su colaborador son amigos, ello servirá de ejemplo. Usted tendría una asociación "es amigo de", así como "es colaborador de", como se aprecia en la figura 2.9.

FIGURA 2.9

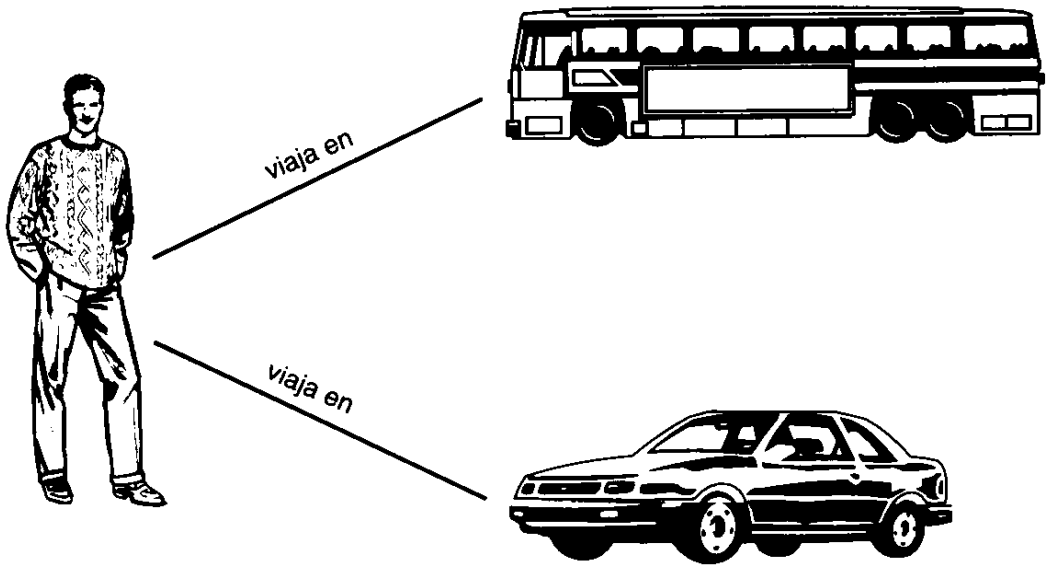
En ocasiones, los objetos pueden asociarse en más de una forma.



Una clase se puede asociar con más de una clase distinta. Una persona puede viajar en automóvil, pero también puede hacerlo en autobús (vea la figura 2.10).

FIGURA 2.10

Una clase puede asociarse con más de una clase distinta.



TÉRMINO NUEVO

La *multiplicidad* (o diversificación) es un importante aspecto de las asociaciones entre objetos. Indica la cantidad de objetos de una clase que se relacionan con otro objeto en particular de la clase asociada. Por ejemplo, en un curso escolar, el curso se imparte por un solo instructor, en consecuencia, el curso y el instructor están en una asociación de uno a uno. Sin embargo, en un seminario hay diversos instructores que impartirán el curso a lo largo del semestre, por lo tanto, el curso y el instructor tienen una asociación de uno a muchos.

Podrá encontrar todo tipo de multiplicidades si echa una mirada a su alrededor. Una bicicleta rueda en dos neumáticos (multiplicidad de uno a dos), un triciclo rueda en tres, y un vehículo de 18 ruedas, en 18.

Agregación

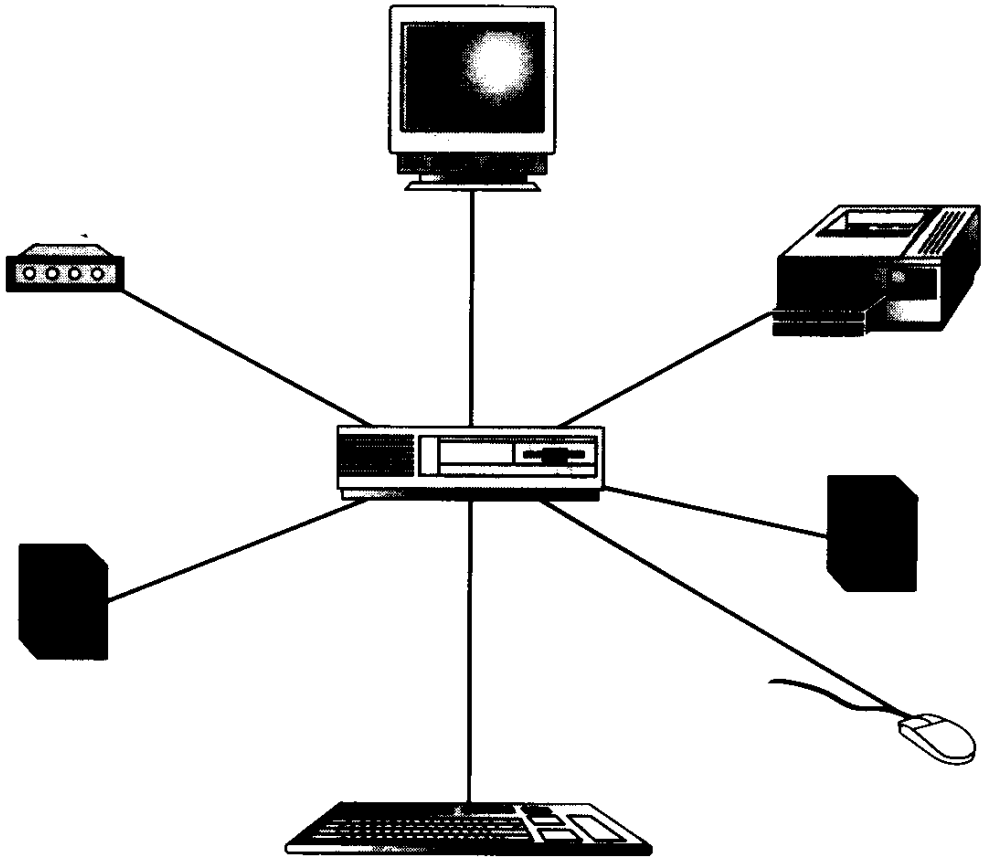
Vea su computadora: cuenta con un gabinete, un teclado, un ratón, un monitor, una unidad de CD-ROM, uno o varios discos duros, un módem, una unidad de disquete, una impresora y, posiblemente, bocinas. Dentro del gabinete, junto con las citadas unidades de disco, tiene una CPU, una tarjeta de vídeo, una de sonido y otros elementos sin los que, sin duda, difícilmente podría vivir.

TÉRMINO NUEVO

Su computadora es una *agregación* o adición, otro tipo de asociación entre objetos. Como muchas otras cosas que valdrían la pena tener, su equipo está constituido de diversos tipos de componentes (vea la figura 2.11). Tal vez conozca varios ejemplos de agregaciones.

FIGURA 2.11

Una computadora es un ejemplo de agregación: un objeto que se conforma de una combinación de diversos tipos de objetos.



TÉRMINO NUEVO

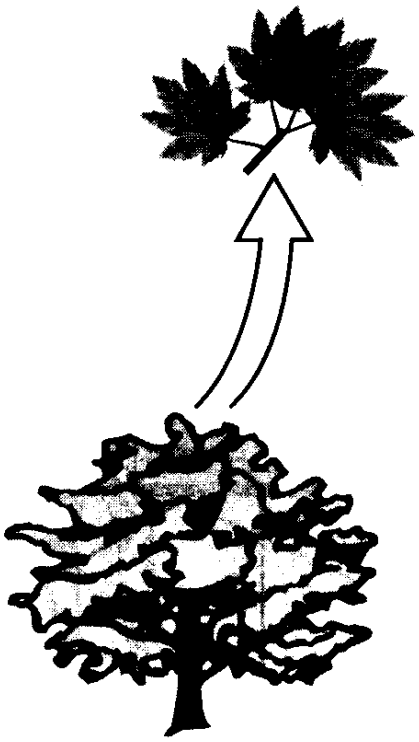
Un tipo de agregación trae consigo una estrecha relación entre un objeto agregado y sus objetos componentes. A esto se le conoce como *composición*. El punto central de la composición es que el componente se considera como tal sólo como parte del objeto compuesto. Por ejemplo: una camisa está compuesta de cuerpo, cuello, mangas, botones, ojales y puños. Suprima la camisa y el cuello será inútil.

En ocasiones, un objeto compuesto no tiene el mismo tiempo de vida que sus propios componentes. Las hojas de un árbol pueden morir antes que el árbol. Si destruye al árbol, también las hojas morirán (vea la figura 2.12).

La agregación y la composición son importantes dado que reflejan casos extremadamente comunes, y ello ayuda a que cree modelos que se asemejen considerablemente a la realidad.

FIGURA 2.12

En una composición, un componente puede morir antes que el objeto compuesto. Si destruye al objeto compuesto, destruirá también a los componentes.



La recompensa

Los objetos y sus asociaciones conforman la columna vertebral de la funcionalidad de los sistemas. Para modelarlos, necesitará comprender lo que son las asociaciones. Si está consciente de los posibles tipos de asociaciones, tendrá una amplia gama de posibilidades cuando hable con los clientes respecto a sus necesidades, obtendrá sus requerimientos y creará los modelos de los sistemas que los ayudarán a cumplir con sus retos de negocios.

TERMINO NUEVO

Lo importante es utilizar los conceptos de la orientación a objetos para ayudarle a comprender el área de conocimiento de su cliente (su *dominio*), y esclarecer sus puntos de vista al cliente en términos que él o ella puedan comprender.

Es allí donde se aplica el UML. En las siguientes tres horas, aprenderá a aplicar el UML para visualizar los conceptos que ha aprendido durante esta hora.

Resumen

La orientación a objetos es un paradigma que depende de algunos principios fundamentales. Un objeto es una instancia de una clase. Una clase es una categoría genérica de objetos que tienen los mismos atributos y acciones. Cuando crea un objeto, el área del problema en que trabaje determinará cuántos de los atributos y acciones debe tomar en cuenta.

La herencia es un aspecto importante de la orientación a objetos, un objeto hereda los atributos y operaciones de su clase. Una clase también puede heredar atributos y acciones de otra.

El polimorfismo es otro aspecto importante, ya que especifica que una acción puede tener el mismo nombre en diferentes clases y cada clase ejecutará tal operación de forma distinta.

Los objetos ocultan su funcionalidad de otros objetos y del mundo exterior. Cada objeto presenta una interfaz para que otros objetos (y personas) puedan aprovechar su funcionalidad.

Los objetos funcionan en conjunto mediante el envío de mensajes entre ellos. Los mensajes son peticiones para realizar operaciones.

Por lo general, los objetos se asocian entre sí y esta asociación puede ser de diversos tipos. Un objeto en una clase puede asociarse con cualquier cantidad de objetos distintos en otra clase.

La agregación es un tipo de asociación. Un objeto agregado consta de un conjunto de objetos que lo componen y una composición es un tipo especial de agregación. En un objeto compuesto, los componentes sólo existen como parte del objeto compuesto.

Preguntas y respuestas

P **Usted dijo que la orientación a objetos ha tomado por asalto al mundo del software. ¿Qué no hay algunas aplicaciones importantes que no están orientadas a objetos?**

R Sí, y se conocen como sistemas "heredados" (programas que en muchos casos son ejecutados para mostrar su época). La orientación a objetos ofrece diversas ventajas, como la reusabilidad y un rápido periodo de desarrollo. Por tales razones, muy probablemente verá las nuevas aplicaciones (y las versiones rediseñadas de varias aplicaciones antiguas) escritas bajo el esquema de la orientación a objetos.

Taller

Para repasar lo que ha aprendido de la orientación a objetos, intente responder a algunas preguntas y realizar los siguientes ejercicios. Las respuestas las encontrará en el Apéndice A, "Respuestas a los cuestionarios".

Cuestionario

1. ¿Qué es un objeto?
2. ¿Cómo trabajan los objetos en conjunto?
3. ¿Qué establece la multiplicidad?
4. ¿Pueden asociarse dos objetos entre sí en más de una manera?

Ejercicios

Esta es una hora teórica, así que no incluí ejercicios. Verá algunos en las siguientes horas.