

# Tema 1

La naturaleza e importancia  
de los requerimientos



# Desarrollo propio



## Introducción

El presente material es un compendio de información dirigido a los estudiantes que matriculan el curso de la Ingeniería de Requerimientos, el cual, se encuentra ubicado en el bloque VII del plan de estudios del Bachillerato de la Escuela de Informática de la Universidad Estatal a Distancia (UNED).

Este material didáctico tiene como objetivo ofrecer a los estudiantes los principios básicos de la Ingeniería de Requerimientos a partir de información actualizada de diferentes autores.

El libro está estructurado a partir de los temas básicos de la ingeniería de requerimientos, los cuales son:

- La naturaleza e importancia de la ingeniería de requerimientos
- Introducción a la ingeniería de requerimientos
- El proceso de ingeniería de requerimientos
- Análisis de sistemas
- Técnicas de exploración y análisis de los requerimientos
- Especificación de requerimientos con casos de uso

La ingeniería de software es una disciplina compuesta por diversas actividades. Una de esas actividades es la ingeniería de requisitos o requerimientos, la cual, tiene como objetivo principal determinar de manera óptima las necesidades de los usuarios de software antes de entrar en la fase de diseño.

Para la ingeniería de requerimientos, la tarea más importante que el ingeniero de software o analista de sistemas realiza para el cliente (futuros usuarios del sistema) es la extracción detallada y refinada de los requerimientos del software a construir o mantener.

Para que los estudiantes sean capaces de comprender lo anterior, inicialmente se presenta el concepto de la ingeniería de requerimientos, las características de los requerimientos y las diversas dificultades que tanto el analista de sistemas o el ingeniero de software pueden enfrentar para definirlos de manera óptima.

Seguidamente, se procede a explicar cómo definir los interesados e involucrados en el desarrollo del software. Además, explica como identificar las necesidades y cómo modelar el dominio. En este apartado también se define qué es un requerimiento funcional y qué un requerimiento no funcional.

En el tercer tema se procede a explicar el proceso de ingeniería de software y cómo realizar cada una de sus actividades: extracción, análisis, especificación, validación y administración de los requerimientos.

El tema cuatro corresponde al análisis y modelado de sistemas, el cual puede ser abordado desde diferentes enfoques compuesto por diferentes fases: análisis del problema, definición del alcance y diseño lógico de los requerimientos.

En el tema cinco se describen diferentes técnicas de extracción de requerimientos como los son: las entrevistas, investigación y visitas a sitio, prototipos y la planeación conjunta.

Para finalizar, se aborda el tema de descripción formal de los requerimientos a partir de la técnica de casos de uso, la cual, describe de manera detallada el comportamiento de un sistema.



# Tema 2

Introducción a la Ingeniería  
de Requerimientos

Comprender la tipificación  
de requerimientos y bases  
generales del proceso de  
Ingeniería de Requerimientos



# 1 El contexto de métodos de análisis y diseño de sistemas

## Panorámica y objetivos del capítulo

Éste es un libro acerca del análisis y diseño de sistemas como se aplican a los sistemas de información y a las aplicaciones de cómputo. Sin importar su ocupación o puesto en cualquier empresa, probablemente usted participará en un análisis y diseño de sistemas. Algunos de ustedes se volverán *analistas de sis temas*, los jugadores fundamentales en el análisis de sistemas y las actividades de diseño. El resto de ustedes trabajará *con analistas de sistemas* conforme los proyectos vayan y vengan en sus organizaciones. En este capítulo se presentan los sistemas de información desde cuatro perspectivas distintas. Usted comprenderá el contexto del análisis de sistemas y los métodos de diseño, donde podrá:

- Definir *sistema de información* y nombrar siete tipos de sistemas de información.
- Identificar distintos tipos de *involucrados* que utilicen o desarrollen sistemas de información y dar ejemplos de cada uno.
- Definir el papel único de los *analistas de sistemas* en el desarrollo de sistemas de información.
- Identificar aquellas *habilidades* necesarias para funcionar con éxito como analista de sistemas de información.
- Describir los *impulsores de negocios* actuales que influyen en el desarrollo de sistemas de información.
- Describir los *impulsores de tecnología* actuales que influyen en el desarrollo de sistemas de información.
- Describir brevemente un *proceso simple* para desarrollar sistemas de información.

## Introducción

Es la primera semana de trabajo de Bob Martínez como analista/programador. Recién salido de la universidad con un título en tecnología en sistemas de información de cómputo, está ansioso por trabajar con sistemas de información en el mundo real. Su empleador es SoundStage Entertainment Club, uno de los clubes de música y video de mayor crecimiento en Estados Unidos. SoundStage apenas comienza un trabajo de análisis y diseño de sistemas en una reingeniería de sus servicios de membresía de sistemas de información. Bob ha sido asignado al equipo de proyecto.

Esta mañana fue la junta de inicio para el proyecto, una reunión que incluyó al vicepresidente de servicios de membresía, al director del club de audio, al director del club de juegos, al director de marketing, al director de servicio al cliente y al director de las operaciones de almacén. Con esa alineación, Bob estuvo contento de mantenerse en silencio en la junta y de confiar en su jefa, Sandra Shepherd, una analista de sistemas *senior*. Él estaba sorprendido de lo bien que Sandra era capaz de hablar el lenguaje de cada uno de los participantes y explicar los planes para el nuevo sistema de información, en términos que ellos pudieran entender y con los beneficios que pudieran apreciar. Bob había pensado que al haber apenas salido de la universidad él sabría más acerca de la tecnología de punta que la mayoría de sus colaboradores. Pero Sandra parecía entenderlo todo acerca del comercio electrónico y el uso de tecnologías móviles, además de muchas cosas de las que Bob sólo tenía un vago conocimiento. Él hizo una nota para leer acerca de los sistemas ERP, cuando salieron en la discusión. Al final de la junta Bob tenía una nueva apreciación del puesto de analista de sistemas y de todas las cosas que aún tenía que aprender.

**sistema** Grupo de componentes interrelacionados que funcionan juntos para lograr un resultado deseado.

## Marco de referencia para análisis y diseño de sistemas

**sistema de información (IS)** Conjunto de personas, datos, procesos y **tecnología de la información** que interactúan para recopilar, procesar, guardar y proporcionar como salida la información necesaria para brindar soporte a una organización.

**tecnología de información (TI)** Término contemporáneo que describe la combinación de la tecnología de computadoras (hardware y software) con la de telecomunicaciones (redes de datos, imágenes y voz).

**sistema de procesamiento de transacciones (TPS)** Sistema de información en el que se capturan y procesan los datos relativos a transacciones de negocios.

**sistema de información administrativa (MIS)** Sistema de información que provee informes orientados a la administración basado en el procesamiento de las transacciones y operaciones de la organización.

Como el título lo sugiere, este es un libro acerca de *métodos de análisis y diseño de sistemas*. En este capítulo, presentaremos el tema por medio de un marco de referencia visual simple pero completo. Cada capítulo en este libro comienza con una *página de inicio* (vea la página 4) en la que se muestra rápida y visualmente qué aspectos del marco de referencia total analizaremos en el capítulo. Construiremos este marco de referencia visual lentamente sobre los primeros tres capítulos para evitar inundarlo tan pronto con demasiados detalles. A partir de ahí, cada capítulo resaltará los aspectos del marco de referencia que son enseñados con mayor detalle dentro de ese capítulo.

Finalmente, este es un libro que trata acerca de “analizar” los requerimientos de negocios para los sistemas de información y “diseñar” los *sistemas de información* que satisfagan esos requerimientos de negocios. En otras palabras, el *producto* del análisis y diseño de sistemas es un sistema de información. Ese producto está representado visualmente en el marco de referencia visual como el rectángulo grande en el centro de la imagen.

Un **sistema** es un grupo de componentes interrelacionados que funcionan juntos para lograr un resultado deseado. Por ejemplo, usted puede ser propietario de un sistema de teatro en casa conformado por un aparato de DVD, un receptor, bocinas y el monitor.

Los **sistemas de información (IS)**, por sus siglas en inglés) en las organizaciones capturan y administran datos para producir información útil que respalda a una organización y sus empleados, clientes, proveedores y socios. Muchas organizaciones consideran que los sistemas de información son esenciales para su capacidad de competir u obtener una ventaja competitiva. La mayoría de las organizaciones se han percatado de que *todos* los trabajadores necesitan participar en el desarrollo de sistemas de información. Por tanto, el desarrollo de sistemas de información es un tema relevante para usted sin importar si estudia o no para convertirse en un profesional de sistemas de información.

Los sistemas de información vienen en todas formas y tamaños. Están tan entrelazados en la tela de los sistemas de negocios que respaldan que con frecuencia es difícil distinguir entre sistemas de negocios y sus sistemas de información de soporte. Basta con decir que los sistemas de información pueden ser clasificados de acuerdo con las funciones que atienden. Los **sistemas de procesamiento de transacciones (transaction processing systems, TPS)** procesan transacciones de negocios como pedidos, tarjetas de tiempo, pagos y reservaciones. Los **sistemas de información administrativa (management information systems, MIS)** utilizan los datos de transacción para producir información necesaria por los administradores para dirigir el negocio.

Los **sistemas de soporte de decisiones (decision support systems, DSS)** ayudan a diversos tomadores de decisiones a identificar y elegir entre opciones o decisiones. Los **sistemas de información ejecutiva (executive information systems, EIS)** están adaptados a las necesidades de información únicas de los ejecutivos que planean el negocio y evalúan el desempeño contra esos planes. Los **sistemas expertos** capturan y reproducen el conocimiento de un solucionador de problemas experto o un tomador de decisiones y luego simulan el “pensamiento” de ese experto. Los **sistemas de comunicación y colaboración** resaltan la comunicación y la colaboración entre las personas, tanto internas como externas de la organización. Finalmente, los **sistemas de automatización de oficina** ayudan a los empleados a crear y compartir documentos que respaldan las actividades diarias de oficina.

Como se ilustró en la página de inicio del capítulo, los sistemas de información pueden ser vistos desde diversas perspectivas, que incluyen:

- Los jugadores en el sistema de información (el “equipo”).
- Los impulsores de negocios que influyen en el sistema de información.
- Los impulsores de tecnología utilizados por el sistema de información.
- El proceso utilizado para desarrollar el sistema de información.

Examinemos cada una de estas perspectivas en las secciones restantes del capítulo.

## Los jugadores y los involucrados en el sistema

Supongamos que usted está en una posición para ayudar a construir un sistema de información. ¿Quiénes son los **involucrados** en este sistema? Los involucrados en los sistemas de información pueden ser clasificados ampliamente en los cinco grupos que se muestran en el lado izquierdo de la figura 1.1. Nótese que cada grupo de involucrados tiene una perspectiva diferente del mismo sistema de información. El *analista de sistemas* es un involucrado único en la figura 1.1. El analista de sistemas sirve como un facilitador o instructor, que construye puentes entre las brechas de comunicación que pueden desarrollarse en forma natural entre los propietarios del sistema no técnico y los usuarios así como los diseñadores y constructores del sistema técnico.

Todos los involucrados anteriores tienen una cosa en común: son lo que el departamento de trabajo estadounidense llama **trabajadores de la información**. La vida de los trabajadores de la información depende de las decisiones tomadas con base en la información. Actualmente, más del 60 por ciento de la fuerza laboral estadounidense participa en la producción, distribución y uso de la información. Examinemos los cinco grupos de trabajadores de la información con mayor detalle.

Analicemos brevemente las perspectivas de cada grupo. Pero antes, debemos señalar que estos grupos en realidad definen “papeles” jugados en el desarrollo de sistemas. En la práctica, cualquier persona puede tener más de uno de estos papeles. Por ejemplo, un propietario de sistemas podría también ser un usuario de sistemas. En forma similar, un analista de sistemas puede también ser visto como un diseñador de sistemas y un diseñador de sistemas puede ser visto como un constructor de sistemas. Cualquier combinación puede funcionar.

### > Propietarios de sistemas

Para cualquier sistema de información, grande o pequeño, habrá uno o más **propietarios del sistema**. Los propietarios del sistema comúnmente vienen de las filas de los administradores. Para los sistemas de información de medianos a grandes, los propietarios del sistema son, en general, administradores medios o ejecutivos. Para los sistemas pequeños, los propietarios del sistema pueden ser administradores medios o supervisores. Los propietarios del sistema tienden a estar involucrados en la línea de fondo; ¿Cuánto costará el sistema? ¿Cuánto valor o qué beneficios el sistema retornarán al negocio? El valor y los beneficios pueden ser medidos de distintas formas, como se señala en la lista de comprobación del margen.

### > Usuarios de sistemas

Los **usuarios del sistema** constituyen la vasta mayoría de los trabajadores de la información en cualquier sistema de información. A diferencia de los propietarios del sistema, los usuarios del sistema tienden a estar menos preocupados con los costos y los beneficios del

**sistema de soporte de decisiones (DSS)** Sistema de información que ayuda a identificar oportunidades de toma de decisiones o proporciona información que ayuda a tomarlas.

**sistema de información ejecutiva (EIS)** Sistema de información que brinda soporte a las necesidades de planeación y evaluación de los administradores de nivel ejecutivo.

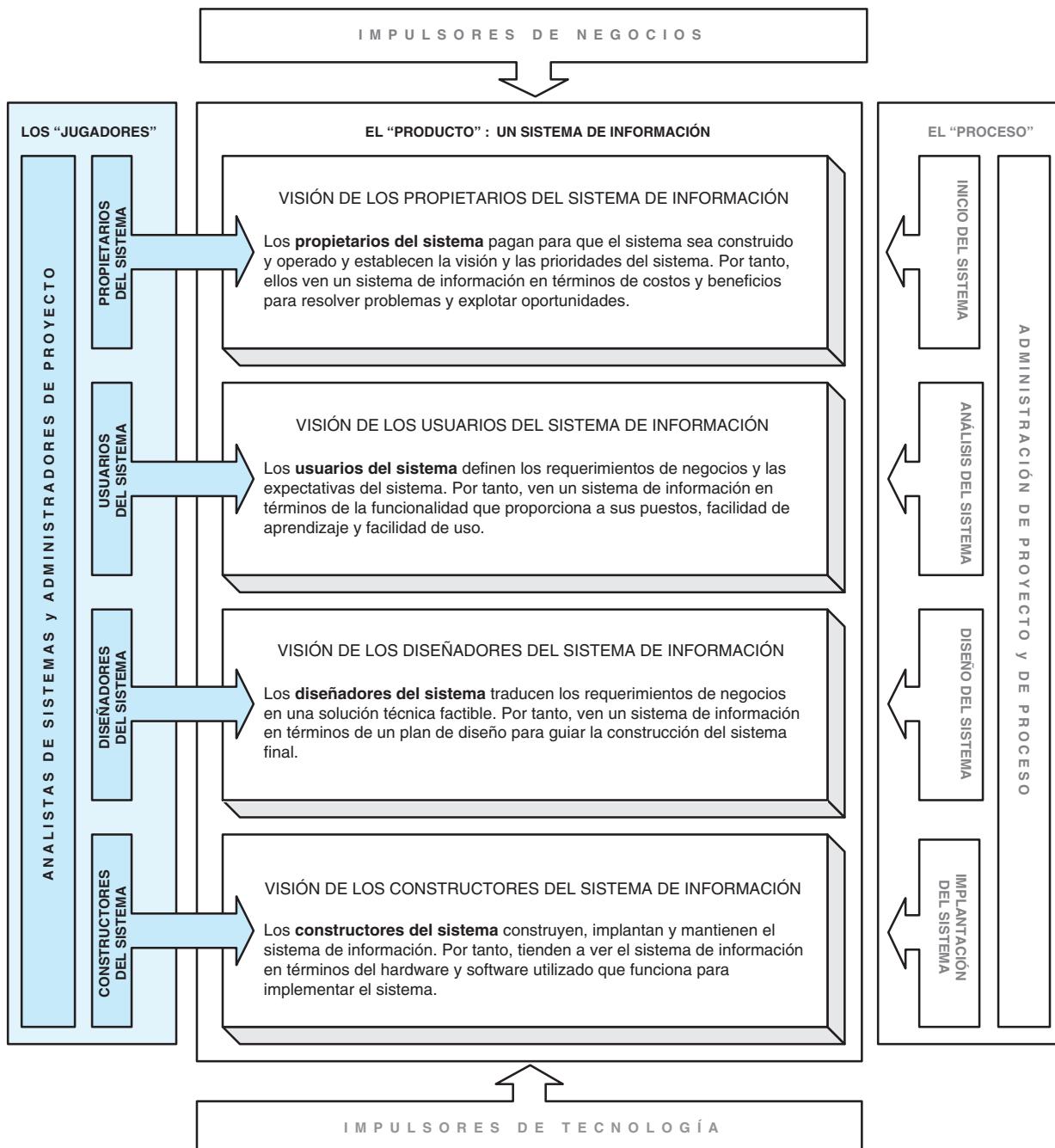
**sistema experto** Sistema de información en el cual se captura la experiencia de los expertos humanos y luego simula esa experiencia para beneficio de quienes no son expertos.

**sistema de comunicación y colaboración** Sistema de información que posibilita la comunicación más efectiva entre los empleados, socios, clientes y proveedores, para mejorar su capacidad de colaboración.

**sistema de automatización de oficina** Sistema de información que brinda soporte a la amplia gama de actividades de oficina de los negocios para mejorar el flujo de trabajo entre los empleados.

**involucrado** Toda persona que tiene interés en un sistema de información existente o propuesto. Los involucrados y grupos de interés pueden ser trabajadores técnicos y no técnicos. También puede tratarse de trabajadores internos y externos.

**trabajador de la información** Toda persona cuyo trabajo entraña la creación, recopilación, procesamiento, distribución y uso de información.



**FIGURA 1.1** Perspectiva de los involucrados sobre un sistema de información

#### proprietario del sistema

Patrocinador y representante ejecutivo de un sistema de información, que generalmente se encarga del financiamiento del proyecto y del desarrollo, operación y mantenimiento del sistema de información.

sistema. En lugar de eso, como se ilustra en la figura 1.1, están preocupados por la funcionalidad que el sistema provee a sus puestos y, la facilidad de aprendizaje y uso del sistema. Aunque los usuarios se han vuelto más cultos en cuanto a la tecnología con el paso de los años, su principal preocupación es que el trabajo se realice. En consecuencia, las discusiones con la mayoría de los usuarios necesitan mantenerse al nivel de los requerimientos de negocios en contraste con el nivel de requerimientos técnicos. Gran parte de este libro está dedicada a enseñarle la forma de identificar y comunicar eficazmente los requerimientos de negocios para un sistema de información.

Existen muchas clases de usuarios de sistemas. Cada clase debe participar directamente en cualquier proyecto de desarrollo de sistemas de información que los afecte. Analicemos brevemente estas clases.

**Usuarios internos del sistema** Los usuarios internos del sistema son empleados del negocio para el cual se construyen la mayoría de los sistemas de información. Los usuarios internos constituyen el mayor porcentaje de usuarios de sistemas de información en la mayoría de las empresas. Los ejemplos incluyen:

- **Trabajadores de oficina y de servicio.** Desempeñan la mayoría del proceso de transacción diaria en el negocio promedio. Procesan pedidos, facturas, pagos y demás. Manejan y archivan la correspondencia. Llenan pedidos en el almacén. Fabrican productos en el piso del taller. La mayoría de los datos fundamentales en cualquier negocio son capturados o creados por estos trabajadores, muchos de los cuales desempeñan una labor manual además del proceso de datos. Los sistemas de información que se dirigen a estos trabajadores, generalmente se enfocan en la velocidad y precisión del proceso de transacciones.
- **Personal técnico y profesional.** Consiste en gran medida de especialistas de negocios e industriales que desempeñan un trabajo que requiere grandes habilidades y especialización. Ejemplos de esto pueden incluir abogados, contadores, ingenieros, científicos, analistas de mercado, diseñadores de publicidad y estadistas. Como su trabajo está basado en ramas del conocimiento bien definidas, a veces son llamados **trabajadores del conocimiento**. Los sistemas de información que se dirigen al personal técnico y profesional se enfocan en el análisis de datos así como en generar información oportuna para la solución de problemas.
- **Supervisores, administradores medios y administradores ejecutivos.** Son los tomadores de decisiones. Los supervisores tienden a enfocarse en la solución de problemas y en la toma de decisiones diaria. Los administradores intermedios están más preocupados por los problemas operacionales (corto plazo) y en la toma de decisiones. Los administradores ejecutivos están preocupados por la planeación y toma de decisiones estratégicas (a largo plazo). Los sistemas de información para los administradores tienden a enfocarse completamente en el acceso a la información. Los administradores necesitan el derecho a la información en el momento correcto para identificar y resolver problemas así como para tomar buenas decisiones.

**Usuarios externos del sistema** Internet ha permitido que las fronteras tradicionales de los sistemas de información se extiendan para incluir otros negocios o consumidores directos como usuarios del sistema. Estos usuarios externos del sistema constituyen un porcentaje cada vez más grande de usuarios de sistemas para los sistemas de información modernos. Los ejemplos incluyen:

- **Clientes.** Cualquier organización o individuo que compra nuestros productos y servicios. En la actualidad, nuestros clientes se pueden convertir en usuarios directos de nuestros sistemas de información cuando pueden ejecutar directamente pedidos y transacciones de ventas que antes requerían intervención de un usuario interno. Por ejemplo, si usted compró un producto de la compañía a través de Internet, se convirtió en un usuario externo del sistema de información de ventas de esa empresa. (No hubo necesidad de que un usuario interno por separado del negocio colocara su pedido.)
- **Proveedores.** Cualquier organización de la que nuestra compañía pueda comprar suministros y materia prima. Estos proveedores pueden interactuar de manera directa con los sistemas de información de nuestra compañía para determinar nuestras necesidades de suministros y automáticamente crear pedidos para satisfacer esas necesidades. Ya no hay necesidad de que un usuario inicie esos pedidos a un proveedor.
- **Socios.** Cualquier organización de la que nuestra compañía puede adquirir servicios o con la que se asocia. La mayoría de los negocios modernos contrata o subcontrata diversos servicios básicos como mantenimiento del sitio, administración de redes y muchos más. Y las empresas han aprendido a asociarse con otras para impulsar con mayor rapidez las fortalezas y construir mejores productos con más rapidez.
- **Empleados.** Son aquellos que trabajan en viajes o en su casa. Por ejemplo, los representantes de ventas generalmente pasan gran parte de su tiempo en viajes. También, muchas empresas permiten a los empleados trabajar a distancia (es decir, hacerlo desde casa) para reducir costos y mejorar la productividad. Como usuarios móviles o remotos, estos empleados requieren acceso a los mismos sistemas de información así como los requeridos por los usuarios internos.

## VENTAJAS Y BENEFICIOS DE LOS SISTEMAS DE INFORMACIÓN

Aumento en la utilidad del negocio
Reducción de los costos del negocio
Costos y beneficios del sistema
Aumento en la participación de mercado
Mejora en las relaciones con los clientes
Aumento en la eficiencia
Mejor toma de decisiones
Mejor cumplimiento de la normatividad
Menos errores
Mejor seguridad
Mayor capacidad

### usuario del sistema

"Cliente" que usa con regularidad un sistema de información o se ve afectado por él, capturando, validando, introduciendo, respondiendo, almacenando e intercambiando datos e información.

### trabajador del conocimiento

Todo trabajador cuyas responsabilidades se centran en un área especializada de conocimientos.

**usuario remoto** Usuario que no se ubica físicamente en las instalaciones donde se encuentran los sistemas de información pero que necesita y tiene acceso a ellos.

**usuario móvil** Usuario cuya ubicación cambia constantemente pero requiere acceso a los sistemas de información desde cualquier lugar.

#### diseñador de sistemas

Especialista técnico que traduce los requerimientos de negocios de los usuarios del sistema y las restricciones en soluciones técnicas. Diseña las bases de datos, entradas, salidas, pantallas, redes y software que podrán satisfacer los requerimientos de los usuarios del sistema.

#### > Diseñadores de sistemas

Los **diseñadores de sistemas** son especialistas en tecnología de sistemas de información. Como se muestra en la figura 1.1 estos diseñadores están involucrados en opciones de tecnología de información y en el diseño de sistemas que utilizan tecnologías elegidas. Los diseñadores de sistemas actuales tienden a enfocarse en especialidades técnicas. Algunos de ustedes pueden estarse preparando para una de las siguientes especialidades técnicas, como:

- **Administradores de bases de datos.** Especialistas en tecnologías de bases de datos que diseñan y coordinan cambios a las bases de datos corporativas.
- **Arquitectos de redes.** Especialistas en creación de redes y tecnología de telecomunicaciones que diseñan, instalan, configuran, optimizan y respaldan redes locales y de áreas amplias, que incluyen conexiones a Internet y otras redes externas.
- **Artistas gráficos.** Relativamente nuevos en la mezcla de trabajadores de tecnología de la información (TI) actual, los especialistas en tecnología y métodos gráficos diseñan y construyen interfaces fascinantes y fáciles de utilizar para sistemas, que incluyen interfaces para computadoras personales, la Web, manuales y teléfonos inteligentes.
- **Expertos en seguridad.** Especialistas en la tecnología y los métodos utilizados para garantizar la seguridad (y privacidad) de datos y redes.
- **Especialistas en tecnología.** Expertos en la aplicación de tecnologías específicas que serán utilizadas en un sistema (por ejemplo, un paquete de software comercial específico o un tipo específico de hardware).

#### > Constructores de sistemas

**constructor del sistema** Especialista técnico que construye sistemas de información y sus componentes con base en las especificaciones de diseño que generan los diseñadores de sistemas.

Los **constructores de sistemas** (de nuevo, vea la figura 1.1) son otra categoría de especialistas de tecnología para sistemas de información. Su papel es construir el sistema de acuerdo con las especificaciones de los diseñadores del sistema. En las organizaciones pequeñas o con sistemas de información pequeños, los diseñadores de sistemas y los constructores de sistemas con frecuencia son las mismas personas. Pero en las organizaciones grandes con sistemas de información grandes a menudo ocupan puestos separados. Algunos de ustedes pueden estarse preparando para alguna de las siguientes especialidades técnicas, como:

- **Programadores de aplicaciones.** Especialistas que convierten los requerimientos de negocios y las declaraciones de problemas y procedimientos en lenguajes de computadora. Desarrollan y prueban programas de cómputo para capturar y almacenar datos, localizar y recuperar datos para aplicaciones de cómputo.
- **Programadores de sistemas.** Especialistas que desarrollan, prueban e implementan software a nivel sistema operativo, utilerías de software y otros servicios. Con mayor frecuencia, también desarrollan “componentes” de software reutilizable para uso por parte de programadores de aplicaciones (arriba).
- **Programadores de bases de datos.** Especialistas en lenguajes y tecnología de bases de datos que construyen, modifican y prueban estructuras de bases de datos y los programas que las utilizan y las mantienen.
- **Administradores de red.** Especialistas que diseñan, instalan, prueban contra fallas y optimizan las redes de cómputo.
- **Administradores de seguridad.** Especialistas que diseñan, implementan, prueban contra fallas y manejan los controles de seguridad y privacidad en una red.

- **Webmasters.** Especialistas que codifican y mantienen los servidores Web.
- **Integradores de software.** Especialistas que integran paquetes de software con hardware, redes y otros paquetes de software.

Aunque con este texto no se pretende educar directamente al constructor del sistema, sí se busca enseñar a los diseñadores de sistemas la forma de comunicar mejor las especificaciones de diseño a los constructores del sistema.

## > Analistas de sistemas

Como usted ha visto, los propietarios, usuarios, diseñadores y constructores de sistemas con frecuencia tienen muy distintas perspectivas acerca de cualquier sistema de información que se va a construir y a utilizar. Algunos están involucrados en generalidades, mientras que otros se enfocan a los detalles. Algunos no son técnicos, mientras que otros son muy técnicos. Esto presenta una brecha de comunicación que siempre ha existido entre quienes necesitan soluciones de negocios basadas en computadora y quienes entienden la tecnología de la información. El **analista de sistemas** crea puentes para acortar esa brecha. Usted puede (y probablemente lo hará) tener un papel como analista de sistemas o alguien que trabaja con analistas de sistemas.

Como se ilustró en la figura 1.1, su papel intencionalmente se superpone con los papeles de todos los demás involucrados. Para los propietarios y usuarios del sistema, los analistas de sistemas identifican y validan los problemas y necesidades de negocios. Para los diseñadores y constructores del sistema, los analistas de sistemas se aseguran de que la solución técnica satisfaga las necesidades del negocio e integre la solución técnica dentro del mismo. En otras palabras, los analistas de sistemas *facilitan* el desarrollo de los sistemas de información a través de la interacción con los demás involucrados.

Existen diversas variaciones legítimas pero a menudo confusas en el título del puesto que llamamos “analista de sistemas”. Un *analista/programador* (o *programador/analista*) incluye las responsabilidades del programador de cómputo y del analista de sistemas. Un *analista de negocios* se enfoca sólo en los aspectos no técnicos de un análisis y diseño de sistemas. Otros sinónimos de “analista de sistemas” son consultor de sistemas, analistas de sistemas, arquitecto de sistemas, ingeniero de sistemas, ingeniero de información, analista de información e integrador de sistemas.

Algunos de ustedes se convertirán en analistas de sistemas. Los demás trabajarán rutinariamente con analistas de sistemas que les ayudarán a resolver sus problemas de negocios e industriales al crear y mejorar su acceso a los datos e información necesaria para hacer su trabajo. Demos un vistazo más de cerca a los analistas de sistemas como facilitadores fundamentales del desarrollo de sistemas de información.

**El papel del analista de sistemas** Los analistas de sistemas entienden tanto de negocios como de cómputo. Estudian los problemas y oportunidades del negocio y luego transforman los requerimientos de negocios y de información en especificaciones de sistemas de información que serán implementados por diversos especialistas técnicos que incluyen programadores de computadoras. Los sistemas de computadoras y de información son valiosos para una empresa sólo si ayudan a resolver problemas o si implementan mejoras.

Los analistas de sistemas inician el *cambio* dentro de una organización. Cada nuevo sistema transforma el negocio. Cada vez más, los mejores analistas de sistemas literalmente cambian sus organizaciones, al proporcionar información que puede ser utilizada para tener una ventaja competitiva, al encontrar nuevos mercados y servicios e incluso al cambiar y mejorar dramáticamente la forma en que la organización hace negocios.

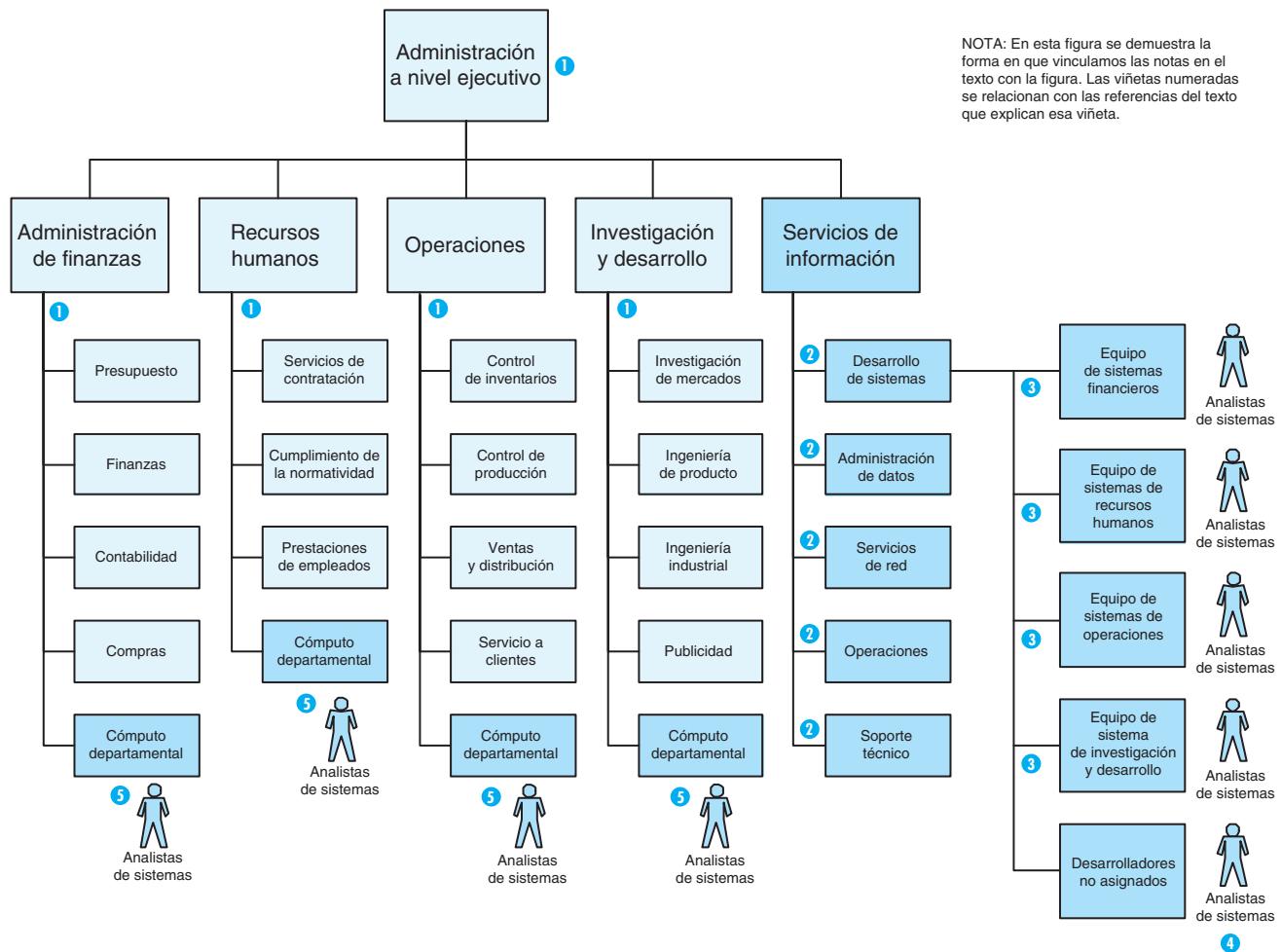
El analista de sistemas es básicamente un *solucionador de problemas*. A través de este texto, el término *problema* será utilizado para describir muchas situaciones, que incluyen:

- Problemas, ya sean reales o anticipados, que requieren de una acción correctiva.
- Oportunidades para mejorar una situación a pesar de la ausencia de quejas.
- Directivas para cambiar una situación sin importar si alguien se ha quejado de la situación actual.

---

**analista de sistemas** Especialista que estudia los problemas y necesidades de una organización para determinar la forma en que las personas, los datos, los procesos y la tecnología de la información pueden lograr óptimamente mejoras para la empresa.

El puesto de analista de sistemas presenta un reto fascinante y emocionante para muchas personas. Ofrece una alta visibilidad administrativa y oportunidades para una



**FIGURA 1.2** Analistas de sistemas en una organización típica

toma de decisiones importante y una creatividad que puede afectar a una organización completa. Es más, este puesto puede ofrecer estos beneficios relativamente temprano en su carrera (comparado al nivel de ingreso de otros trabajos y carreras).

**¿Dónde trabajan los analistas de sistemas?** Cada empresa se organiza de manera única. Pero ciertos patrones de organización parecen ser recurrentes. La figura 1.2 es un organigrama representativo. Las siguientes viñetas numeradas hacen referencia y enfatizan los puntos fundamentales en la figura:

- 1 Los propietarios y usuarios de sistemas están ubicados en las unidades y subunidades funcionales del negocio, así como en la administración ejecutiva.
- 2 Los diseñadores y constructores de sistemas están ubicados generalmente en la unidad de sistemas de información del negocio. La mayoría de los analistas de sistemas también trabaja para la unidad de servicios de información de una organización.
- 3 Como se muestra en la figura, los analistas de sistemas (junto con los diseñadores y constructores de sistemas) pueden estar asignados permanentemente a un equipo que respalda una función de negocios específica (por ejemplo, sistemas financieros).

Los números 2 y 3 anteriores representan un enfoque tradicional para organizar a los analistas de sistemas y otros desarrolladores. Los números 4 y 5 posteriores representan estrategias que tienen la intención de enfatizar la eficiencia o la experiencia de negocios. Todas las estrategias pueden ser combinadas en una sola organización.

## La próxima generación:

### Prospectos de carrera para analistas de sistemas

Muchos de ustedes están considerando o se están preparando para una carrera como analista de sistemas. La vida de un analista de sistemas es desafiante y gratificante. Pero, ¿cuáles son las perspectivas para el futuro? ¿Las organizaciones necesitan analistas de sistemas? ¿Los necesitarán en el futuro inmediato? ¿Ese puesto está cambiando para el futuro?, y si es así, ¿en qué forma? Abordaremos estas preguntas en este recuadro.

De acuerdo con el Departamento de Trabajo estadounidense, los puestos relacionados con cómputo representan cinco de las 20 ocupaciones de más rápido crecimiento en la economía. Lo que es más, estas ocupaciones, relacionadas con cómputo, tienen rápido crecimiento y pagan mejor que muchos otros puestos.

En el 2002, 468 000 trabajadores eran clasificados como analistas de sistemas. Para el 2012, ese número crecerá a 653 000, un aumento de 39%. Esto significa que al menos 185 000 nuevos analistas de sistemas deben ser preparados y contratados (sin incluir a los que requieren ser reemplazados por retiro o por haberse cambiado a posiciones administrativas u otras ocupaciones). La necesidad va en aumento debido a que la industria necesita analistas de sistemas para satisfacer al parecer la infinita demanda por más sistemas de información y aplicaciones de software. Como algunos puestos de programación son subcontratados a contratistas independientes y a otros países, la necesidad aumenta todavía más para analistas de sistemas hábiles, que puedan crear especificaciones de diseño sólidas para equipos de desarrollo remotos. Las oportunidades para el éxito serán mejores para los analistas más preparados, calificados, hábiles y experimentados.

¿Qué sucede con el analista de sistemas exitoso? ¿Un puesto como analista de sistemas lleva a cualquier otra carrera? De hecho, existen muchos caminos de carrera. Algunos analistas de sistemas dejan el campo de sistemas de información y se unen a la comunidad de usuarios. Su experiencia con el desarrollo de aplicaciones de negocios, combinada con su perspectiva de sistemas integral, puede hacer que los analistas de sistemas experimentados sean especialistas de negocios. Alternativamente, los analistas de sistemas se pueden volver administradores de proyectos, administradores de sistemas de información o especialistas técnicos (de bases de datos, telecomunicaciones, microcomputadoras y demás).

Finalmente, los analistas de sistemas hábiles a menudo son reclutados por las industrias de consultoría y contratación externa. Las oportunidades de caminos de carrera son virtualmente ilimitadas.

Como en cualquier profesión, los analistas de sistemas pueden esperar cambios. Aunque siempre es peligroso pronosticar cambios, lo intentaremos. Creemos que las organizaciones se volverán cada vez más dependientes de las fuentes externas para sus analistas de sistemas, consultores y contratistas externos. Esto será dirigido por factores tales como la complejidad y el cambio rápido de tecnología, el deseo de acelerar el desarrollo de sistemas, y la dificultad continua en reclutar, retener y volver a entrenar analistas de sistemas hábiles (y otros profesionales de tecnología de la información). En muchos casos, los analistas de sistemas empleados manejarán proyectos a través de contratos de consultoría o subcontratación.

Creemos que un porcentaje cada vez mayor de los analistas de sistemas de mañana no trabajarán en el departamento de sistemas de información. En lugar de eso, trabajarán en forma directa para una unidad de negocios dentro de una organización. Esto les permitirá atender mejor a sus usuarios. También dará a los usuarios mayor poder sobre qué sistemas son construidos y respaldados.

Finalmente, también creemos que un mayor porcentaje de analistas de sistemas tendrá antecedentes distintos al cómputo. En algún momento, la mayoría de los analistas eran especialistas de cómputo. Actualmente, los graduados en cómputo se vuelven más conocedores en negocios. En forma similar, los recientes graduados en negocios y en carreras no relacionadas con el cómputo se vuelven cada vez más expertos de cómputo. Su ayuda y conocimientos de tiempo completo se necesitarán para satisfacer la demanda y proporcionar los antecedentes de negocios necesarios para las aplicaciones más complejas del mañana.

- ④ Los analistas de sistemas (junto con los diseñadores y constructores de sistemas) pueden también ser compartidos y temporalmente asignados a proyectos específicos para cualquier función de negocios según sea necesario. (Algunas organizaciones creen que este método arroja una mayor eficiencia debido a que los analistas y otros desarrolladores siempre son asignados a los proyectos de mayor prioridad sin importar la experiencia en el área de negocios.)
- ⑤ Algunos analistas de sistemas pueden trabajar para organizaciones de cómputo departamentales más pequeñas que respaldan y reportan a sus propias funciones de negocios específicas. (Algunas organizaciones creen que esta estructura resulta en analistas de sistemas que desarrollan una mayor experiencia en su área de negocios asignada para complementar su experiencia técnica.)

Todas las estrategias anteriores, desde luego, pueden reflejarse dentro de una sola organización.

Sin importar dónde se asignan los analistas de sistemas dentro de la organización, es importante estar conscientes de que se reúnen en *equipos de proyectos*. Éstos deben también incluir una representación apropiada de los demás involucrados que previamente analizamos (propietarios de sistemas, usuarios de sistemas, diseñadores de sistemas, y constructores de sistemas). De acuerdo con eso, enfatizaremos la construcción de equipos y el trabajo en equipo a lo largo de este libro.

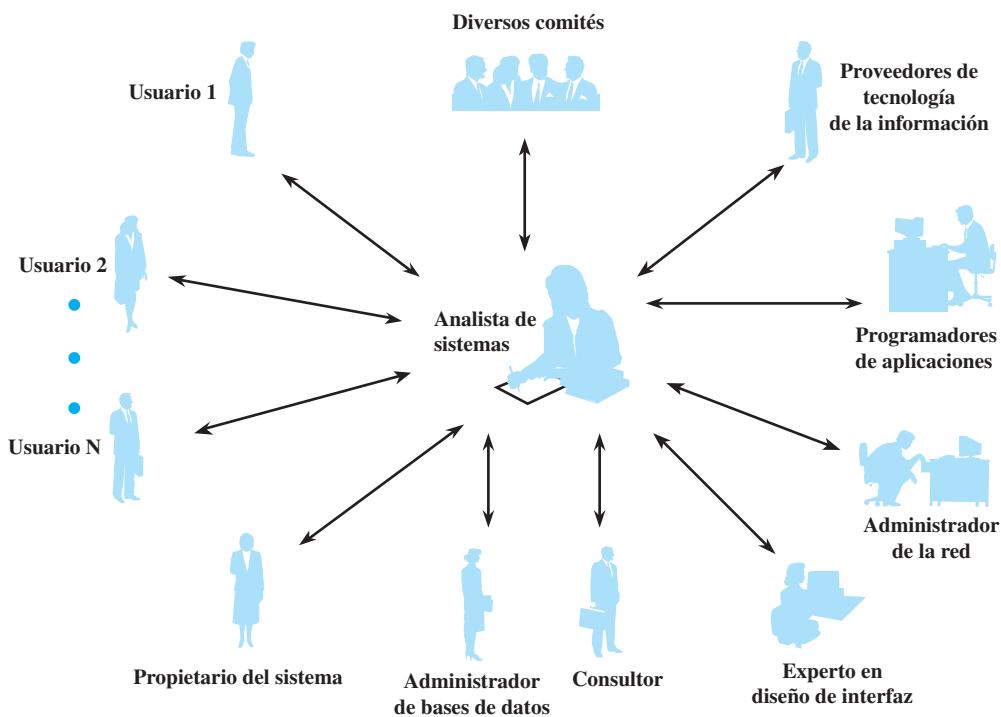
**Habilidades requeridas por el analista de sistemas** Para aquellos de ustedes con aspiraciones de convertirse en analistas de sistemas, en esta sección se describen las habilidades que necesitarán desarrollar. En este libro se presentan muchos análisis de sistemas y conceptos de diseño, herramientas y técnicas. Pero usted también necesitará habilidades y experiencias que ni este libro ni su curso de análisis y diseño pueden proporcionar completamente.

Cuando todo esto falla, el analista de sistemas que recuerda los conceptos básicos y los principios del “pensamiento de sistemas” aún tendrá éxito. No hay herramienta, técnica, proceso o metodología perfecta en todas las situaciones. Pero los conceptos y principios del pensamiento de sistemas siempre le ayudarán a adaptarse a situaciones nuevas y diferentes. En este libro se enfatiza el pensamiento de sistemas.

Hace poco tiempo se pensaba que las únicas herramientas reales de un analista de sistemas eran papel, lápiz y una plantilla de diagrama de flujos. Al paso de los años, varias herramientas y técnicas han sido desarrolladas para ayudar al analista de sistemas. Desafortunadamente, muchos libros enfatizan una clase específica de herramientas que es asociada con una metodología o enfoque al análisis y diseño de sistemas. Conforme usted lea este libro, su caja de herramientas aumentará para incluir muchas de las diferentes metodologías y enfoques para el análisis y diseño de sistemas. Subsecuentemente, usted deberá elegir y utilizar las herramientas con base en las distintas situaciones que encontrará como analista; ¡la herramienta correcta para el trabajo correcto!

Además de tener habilidades formales de análisis y diseño de sistemas, un analista de sistemas debe desarrollar o poseer otras habilidades, conocimientos y rasgos para completar el puesto. Éstos incluyen:

- **Conocimiento laboral de tecnologías de información.** El analista debe estar consciente de las tecnologías de información existentes y las que están en surgimiento. Dicho conocimiento puede ser adquirido en cursos universitarios, seminarios y cursos de desarrollo profesional y programas de capacitación corporativa internos. Los analistas practicantes también se mantienen al día a través de una lectura disciplinada y participación en las sociedades profesionales apropiadas. (Para empezar, véanse las lecturas recomendadas al final de este capítulo y los siguientes.)
- **Experiencia en programación de computadoras.** Es difícil imaginar cómo los analistas de sistemas podrían preparar adecuadamente especificaciones técnicas y de negocios para un programador si ellos no tuvieran alguna experiencia en programación. La mayoría de los analistas de sistemas necesitan estar capacitados en uno o más lenguajes de programación de alto nivel.
- **Conocimiento general de procesos y terminología de negocios.** Los analistas de sistemas deben ser capaces de comunicarse con los expertos de negocios para obtener una comprensión de sus problemas y necesidades. Para el analista, al menos parte de este conocimiento viene sólo mediante la experiencia. Al mismo tiempo, los que aspiran a ser analistas deben aprovechar todas las oportunidades para completar cursos de alfabetización en negocios básicos disponibles en las universidades de negocios. Los cursos importantes pueden incluir contabilidad financiera, administración o contabilidad de costos, finanzas, marketing, administración de manufactura u operaciones, administración de la calidad, economía y leyes de negocios.
- **Habilidades generales de solución de problemas.** El analista de sistemas debe ser capaz de tomar un problema de negocios grande, dividirlo en partes, determinar las causas y efectos del problema, y luego recomendar una solución. Los analistas deben evitar la tendencia a sugerir la solución antes de analizar el problema. Para los que aspiran a ser analistas, muchas universidades ofrecen cursos de filosofía que enseñan habilidades en la solución de problemas, pensamiento crítico y razonamiento. Estas “habilidades suaves o ligeras” servirán también a un analista.



**FIGURA 1.3** El analista de sistemas como facilitador

- **Buenas habilidades de comunicación interpersonal.** Un analista debe ser capaz de comunicarse eficazmente, tanto en forma oral como escrita. Casi sin excepción, sus habilidades de comunicación, y no las técnicas, probarán ser el factor más importante en el éxito o fracaso de su carrera. Estas habilidades se pueden aprender, pero la mayoría de nosotros debemos esforzarnos a obtener ayuda y a trabajar fuerte para mejorarlas. La mayoría las escuelas ofrecen cursos de escritura técnica y de negocios, de comunicación verbal técnica y de negocios, técnicas de entrevista y escucha; todas ellas habilidades útiles para el analista de sistemas. Estas habilidades se enseñan en el capítulo 6.
- **Buenas habilidades de relaciones interpersonales.** Como se ilustró en la figura 1.3, el analista de sistemas interactúa con todos los involucrados en un proyecto de desarrollo de sistemas. Estas interacciones requieren habilidades interpersonales eficaces que permiten al analista tratar con dinámicas de grupo, política de negocios, conflictos y cambio. Muchas escuelas ofrecen cursos de desarrollo de habilidades interpersonales en temas como trabajo en equipo, principios de persuasión, manejo del cambio, del conflicto y liderazgo.
- **Flexibilidad y adaptabilidad.** No hay dos proyectos iguales. Por eso, no hay un enfoque o estándar mágico que sea igualmente aplicable a todos los proyectos. Los analistas de sistemas exitosos aprenden a ser flexibles y a adaptarse a situaciones y retos únicos. Nuestro enfoque de la caja de herramientas antes mencionado tiene la intención de alentar la flexibilidad en el uso de herramientas y métodos de análisis y diseño de sistemas. Pero usted debe desarrollar una actitud de adaptabilidad para utilizar adecuadamente cualquier caja de herramientas.
- **Carácter y ética.** La naturaleza del puesto del analista de sistemas requiere un carácter fuerte y un sentido de lo correcto y lo incorrecto. Los analistas a menudo obtienen acceso a hechos e información sensible o confidencial que no debe ser revelada al público. También, los productos del análisis y diseño de sistemas son considerados generalmente como propiedad intelectual del empleador. Existen varios estándares para la ética de cómputo. Uno de ellos, del Instituto de Ética de Cómputo, es llamado "Los diez mandamientos de la ética de cómputo" y se muestra en la figura 1.4.

**FIGURA 1.4** Ética para los analistas de sistemas**Los diez mandamientos de la ética de cómputo**

1. No utilizará una computadora para dañar a otras personas.
2. No interferirá con el trabajo de cómputo de otras personas.
3. No se entrometerá en los archivos de cómputo de otras personas.
4. No utilizará una computadora para robar.
5. No utilizará una computadora para dar falso testimonio.
6. No copiará o utilizará software registrado por el que no haya pagado.
7. No utilizará los recursos de cómputo de otras personas sin autorización o compensación adecuada.
8. No se apropiará de la producción intelectual de otra persona.
9. Deberá pensar en las consecuencias sociales del programa que usted escribe o el sistema que usted diseña.
10. Siempre deberá usar una computadora en formas que aseguren consideración y respeto para el resto de la humanidad.

Fuente: Instituto de Ética de Cómputo.

**proveedor de servicio externo (ESP)** Un analista, diseñador o constructor de sistemas que vende su conocimiento y experiencia a otras empresas para ayudarlas en la compra, el desarrollo o la integración de sus soluciones de sistemas de información; puede estar afiliado a una organización de consultoría o servicios.

**administrador de proyectos** Profesional experimentado que acepta la responsabilidad de planear, monitorear y controlar proyectos con respecto a un calendario, presupuesto, entregables, satisfacción del cliente, normas técnicas y calidad del sistema.

**> Proveedores de servicio externo**

Aquellos lectores con algo de experiencia de cómputo pueden preguntarse dónde entran los consultores en nuestra taxonomía de los involucrados. No son aparentes inmediatamente en nuestro marco de referencia. Pero ¡están ahí! Cualquiera de los papeles de involucrados puede ser ocupado por trabajadores internos o externos. Los consultores son un ejemplo de un **proveedor de servicio externo (external service provider, ESP)**. La mayoría de los ESP son analistas de sistemas, diseñadores o constructores que son contratados para traer una experiencia especial a un proyecto específico. Ejemplos de esto pueden ser ingenieros en tecnología, ingenieros de ventas, consultores de sistemas, programadores de contratos e integradores de sistemas.

**> El administrador de proyectos**

Hemos presentado la mayoría de los participantes fundamentales en el desarrollo moderno de sistemas de información: propietarios de sistemas, usuarios, diseñadores, constructores y analistas. Debemos concluir enfatizando el hecho de que estos individuos deben trabajar juntos como equipo para construir con éxito sistemas de información y aplicaciones que beneficiarán el negocio. Los equipos requieren de liderazgo. Por esta razón, generalmente uno o más de estos involucrados toma el papel de **administrador de proyectos** para asegurar que los sistemas se desarrollen a tiempo, dentro del presupuesto y con una calidad aceptable. Como se indica en la figura 1.1, la mayoría de los administradores de proyectos son experimentados analistas de sistemas. Sin embargo, en algunas organizaciones, los administradores de proyectos son seleccionados de las filas de lo que hemos llamado “propietarios de sistemas”. Como sea, la mayoría de las organizaciones han aprendido que la administración de proyectos es un papel especializado que requiere de habilidades y experiencia distintivas.

Cada capítulo proporcionará una guía para una instrucción de ritmo individual bajo el título de “Mapa de aprendizaje”. Al reconocer que los distintos alumnos y lectores tienen diferentes antecedentes e intereses, propondremos caminos de aprendizaje adecuados, la mayoría dentro de este texto, pero algunos más allá del alcance de este libro.

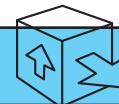
La mayoría de los lectores deben proseguir directamente al capítulo 2 porque los primeros cuatro capítulos proporcionan gran parte del contexto para el resto del libro. Varios temas recurrentes, marcos de trabajo y términos se presentan en esos capítulos para permitirle definir su propio camino de aprendizaje desde ese punto en adelante. En este capítulo nos enfocamos en los sistemas de información desde cuatro distintas perspectivas:

- Los *jugadores*. Desarrolladores y usuarios de los sistemas de información.
- Los *impulsores de negocios* que actualmente influyen en los sistemas de información.
- Los *impulsores de tecnología* que actualmente influyen en los sistemas de información.
- El *proceso* de desarrollo de sistemas de información.

En el capítulo 2 veremos más de cerca al *producto* mismo, los sistemas de información, desde una perspectiva de arquitectura apropiada para el desarrollo de sistemas. Definiremos cómo se visualiza un sistema de información por los distintos jugadores y las etapas de desarrollo.

Más adelante, en el capítulo 3, se analiza de cerca el *proceso* del desarrollo de sistemas.

# Mapa de aprendizaje



## Resumen

1. Los sistemas de información en las organizaciones capturan y administran datos para producir información útil que respalda a una organización y a sus empleados, clientes, proveedores y asociados.
2. Los sistemas de información pueden ser clasificados de acuerdo con las funciones que atienden, lo que incluye:
  - a) Sistemas de proceso de transacción que procesan las transacciones de negocios como pedidos, tarjetas de tiempo (para registrar la entrada), pagos y reservaciones.
  - b) Sistemas de información administrativa que utilizan datos de transacciones para producir información requerida por administradores para dirigir el negocio.
  - c) Sistemas de soporte de decisiones que ayudan a diversos tomadores de decisiones a identificar y elegir entre opciones o decisiones.
  - d) Sistemas de información ejecutiva hechos a la medida para las necesidades de información únicas de ejecutivos que planean el negocio y evalúan el desempeño contra esos planes.
  - e) Sistemas expertos que capturan y reproducen el conocimiento de un solucionador de problemas o un tomador de decisiones experto y luego simulan el “pensamiento” de ese experto.
  - f) Sistemas de comunicación y colaboración que resaltan la comunicación y colaboración entre las personas, tanto internas como externas a la organización.
- g) Sistemas de automatización de oficina que ayudan a los empleados a crear y compartir documentos que respaldan las actividades de oficina diarias.
3. Los sistemas de información pueden ser vistos desde distintas perspectivas, incluida la perspectiva de los “jugadores”, de los “impulsores de negocios” que influyen en el sistema de información, los “impulsores de tecnología” utilizados por el sistema de información y el “proceso” utilizado para desarrollar el sistema de información.
4. Los trabajadores de la información son los involucrados en los sistemas de información. Incluyen a esas personas cuyos puestos abarcan la creación, el acopio, el proceso, la distribución y el uso de la información. Éstos son:
  - a) Propietarios del sistema, patrocinadores y principales defensores de los sistemas de información.
  - b) Usuarios del sistema, la gente que utiliza o que es impactada por el sistema de información en una base regular. Geográficamente, los usuarios del sistema pueden ser internos o externos.
  - c) Constructores del sistema, especialistas de tecnología que construyen el sistema de información basado en las especificaciones de diseño.

- d) Analistas de sistemas, que facilitan el desarrollo de los sistemas de información y las aplicaciones de cómputo. Ellos coordinan los esfuerzos de los propietarios, usuarios, diseñadores y constructores. Con frecuencia, pueden jugar uno de esos papeles también. Los analistas de sistemas realizan el análisis y diseño de sistemas.
5. Además de tener habilidades de análisis y diseño de sistemas formales, un analista de sistemas debe desarrollar o poseer las siguientes habilidades, conocimiento y rasgos:
- a) Conocimiento del trabajo de las tecnologías de la información.
  - b) Experiencia en programación de computadoras.
  - c) Conocimiento general de procesos y terminología de negocios.
  - d) Habilidades generales de solución de problemas.
- e) Habilidades buenas de comunicación interpersonal.
- f) Habilidades buenas de relaciones interpersonales.
- g) Flexibilidad y adaptabilidad.
- b) Carácter y ética.
6. Cualquier papel de involucrado puede ser desempeñado por un trabajador interno o externo llamado proveedor de servicio externo (ESP). La mayoría de los ESP son analistas, diseñadores o constructores de sistemas que son contratados por su experiencia en un proyecto específico.
7. La mayoría de los proyectos de sistemas de información incluyen el trabajo en equipo. Generalmente uno o más de los involucrados (miembros del equipo) toman el papel de administrador de proyecto para asegurarse de que el sistema se desarrolle a tiempo, dentro del presupuesto y con una calidad aceptable. La mayoría de los administradores de proyecto son experimentados analistas de sistemas.

## Preguntas de repaso



1. ¿Por qué los sistemas de información (IS) son esenciales en las organizaciones?
2. ¿Por qué los analistas de sistemas necesitan conocer quiénes son los involucrados en la organización?
3. ¿Quiénes son los involucrados típicos en un sistema de información? ¿Cuáles son sus papeles?
4. Favor de explicar cuáles son las consecuencias si un sistema de información carece de un propietario del sistema.
5. ¿Cuáles son las diferencias entre los usuarios internos y los externos? Dé ejemplos.

6. ¿Cuáles son las diferencias entre el papel del analista de sistemas y el del resto de los involucrados?
7. ¿Qué tipo de conocimientos y habilidades debe poseer un analista de sistemas?
8. Además del conocimiento del negocio y de cómputo que los analistas de sistemas deben poseer, ¿cuáles son las otras habilidades esenciales que necesitan para cumplir eficazmente con su trabajo?
9. ¿Por qué las buenas habilidades de comunicación interpersonales son esenciales para los analistas de sistemas?

## Problemas y ejercicios



1. Suponga que usted es un analista de sistemas que conducirá un análisis de requerimientos para una tienda de ventas tradicional de propiedad individual con un sistema de punto de ventas. Identifique quiénes serían los usuarios típicos internos y externos.
2. Suponga que usted es un analista de sistemas de una empresa consultora y se le ha pedido ayudar al presidente y director ejecutivo de un banco regional. El banco recientemente implementó un plan para reducir la cantidad de personal, incluidos ejecutivos de préstamos, como una estrategia para mantener la rentabilidad. Posteriormente, el banco experimentó problemas crónicos con las solicitudes de préstamo retrasadas debido al número limitado de empleados de préstamos que eran capaces de revisar y aprobar o desaprobar los préstamos. El presidente y director ejecutivo del banco

- está involucrado en soluciones que permitirían que el proceso de aprobación se moviera con mayor rapidez sin aumentar el número de empleados de préstamos y ha contratado a su compañía para obtener soluciones. ¿Cuál es un tipo de sistema que usted podría recomendar al banco?
3. ¿Cómo los sistemas de comunicación y colaboración mejoran la eficiencia y la eficacia? ¿Cuáles son algunos de los sistemas de comunicación y colaboración utilizados por un número cada vez mayor de organizaciones?
  4. Identifique el tipo de sistema de información que los empleados de oficina en una organización utilizarían generalmente y por qué.
  5. Conforme los sistemas de información aumentan en complejidad y profundidad, también aumentan los temas éticos en relación con el acceso y el uso

- de datos de estos sistemas. ¿Cuáles son algunos de estos temas éticos?
6. Aunque los procesos de desarrollo de sistemas y las metodologías pueden variar en gran medida, identifique y explique brevemente las fases “genéricas” del proceso de desarrollo del sistema que son descritas en el libro y que deben ser completadas para cualquier proyecto. Usted es un contratista con una compañía de integración de sistemas.
  7. Identifique a qué fase del proceso de desarrollo pertenecen las siguientes actividades:
- a) Desarrollo del plano técnico o documento de diseño.
  - b) Programación de proyecto.
  - c) Pruebas de integración.
  - d) Entrevistar a los usuarios del sistema para definir los requerimientos del negocio.
8. ¿Cuáles son las dos ventajas más importantes de las tecnologías de software orientadas a objetos sobre las tecnologías de software estructuradas?



## Proyectos e investigación

1. Investigue los salarios promedio de los profesionales de TI. Usted puede utilizar una diversidad de métodos para encontrar esta información, tales como búsqueda en la Web para sitios en línea que publican los resultados de encuestas salariales de profesionales de TI. También puede ver los anuncios clasificados en periódicos, revistas comerciales o en línea.
  - a) ¿Hay una diferencia significativa entre los salarios típicos de analistas, diseñadores y desarrolladores?
  - b) A grandes rasgos, ¿cuál es la diferencia en los salarios típicos de estos distintos grupos?
  - c) ¿Cuáles cree usted que son las razones para esta diferencia?
  - d) ¿Hay una brecha de género en los salarios de los profesionales de TI? Analice cualquier tendencia que haya encontrado y sus implicaciones.
2. Contacte a los directores ejecutivos de información (CIO) o a los gerentes de TI principales de diversas organizaciones locales o regionales. Pregunte acerca del proceso o la metodología que utilizan para el desarrollo de sistemas en sus organizaciones y por qué utilizan ese método en particular.
  - a) Describa y compare los distintos métodos que haya encontrado.
  - b) ¿Cuál método cree usted que sea el más eficaz?
  - c) ¿Por qué?
3. Opciones de carrera y habilidades personales:
  - a) En este punto en su educación, si usted tuviera que escoger entre convertirse en analista de sistemas, diseñador de sistemas o constructor de sistemas, ¿cuál elegiría?
  - b) ¿Por qué?
  - c) Ahora divida un pedazo de papel en dos columnas. En un lado, liste las habilidades personales y los rasgos que usted cree que son los más importantes para cada uno de estos tres
- grupos de analistas, diseñadores y constructores de sistemas. En la segunda columna, liste al menos cinco habilidades y rasgos que usted siente que serán los más fuertes, luego haga un mapa de ellos con las habilidades y rasgos que listó para cada uno de los tres grupos. ¿Con qué grupo tendría usted habilidades y rasgos en común?
- d) ¿Es este grupo el mismo que usted elegiría en la pregunta 3a? ¿Por qué cree usted que este es (o no es) el caso?
4. La biblioteca de su escuela debe tener diarios y periódicos que datan de hace varias décadas o puede suscribirse a servicios de investigación en línea que lo hacen. Mire varios artículos recientes en diarios de tecnología de la información relacionados con análisis de sistemas, así como varios artículos de hace al menos 25 años.
  - a) Compare los artículos recientes con los viejos. ¿Parece haber diferencias significativas en el conocimiento, habilidades, capacidades o experiencia típicos que los analistas de sistemas necesitaban hace 25 años en comparación con la actualidad?
  - b) Si usted encontró diferencias, ¿cuáles son las que considera más importantes?
  - c) ¿Cuáles cree usted que son las razones para estos cambios?
  - d) Ahora saque su bola de cristal y vea al futuro dentro de 25 años. ¿Qué diferencias pronostica usted entre los analistas de sistemas actuales y los de dentro de 25 años?
5. Busque en la Web diversos artículos e información acerca de los temas éticos relacionados con profesionales de la tecnología de la información.
  - a) ¿Qué artículos encontró?
  - b) Con base en su investigación, ¿qué temas éticos cree usted que los analistas de sistemas podrían enfrentar durante sus carreras?

- c) Elija un tema ético en particular y explique los pasos que usted tomaría si fuera confrontado con este tema.
- d) ¿Qué haría usted si encontrara que su mejor amigo y compañero de trabajo ha cometido una seria violación ética? Los hechos que se deben considerar son: La violación puede no ser descubierta nunca, pero le costaría a su compañía muchos miles de dólares en costos más altos durante los próximos años. Su compañía tiene una política estricta de despedir a empleados que cometan violaciones éticas serias. Asegúrese de explicar sus razones para las acciones que usted tomaría, si fuera el caso.
6. Busque en la Web o periódicos de negocios en su biblioteca como *Forbes Magazine* para información

acerca de tres o cuatro ejecutivos de información de compañías u organizaciones grandes.

- a) ¿Qué sector, compañías y directores de la industria encontró usted?
- b) Para cada CIO que usted buscó, ¿cuál fue su experiencia predominante antes de convertirse en CIO; es decir, tenía algún antecedente de tecnología de información, antecedentes de negocios o ambos?
- c) Para cada CIO, ¿cuál fue su nivel de educación?
- d) ¿Cuántos años ha sido cada uno de ellos CIO y aproximadamente en cuántas compañías ha trabajado cada uno?
- e) Con base en su investigación, ¿qué conocimientos y habilidades necesita un CIO con el fin de ser exitoso y por qué?

## Casos breves



1. ¿Qué cree usted que será posible tecnológicamente dentro de 10 años? ¿Y dentro de 20 o 30 años? Investigue una tecnología nueva e interesante que esté en la etapa de investigación y desarrollo. Prepare una presentación con el uso de un clip de película y PowerPoint acerca de esta tecnología y preséntelo a la clase. Emita un ensayo breve acerca de los impactos que esta nueva tecnología podría tener en la sociedad y/o los negocios.
2. Considere las subcontrataciones: Muchas veces se da el caso de que al menos parte del proceso de desarrollo sea subcontratado. De hecho, los líderes de proyecto actuales deben ser capaces de manejar equipos geográficamente diversos así como restricciones de tiempo y de recursos. La subcontratación trae a la mesa el aumento en la eficiencia y ganancias económicas a las sociedades que interactúan. Sin embargo, estas ganancias no son notadas rápidamente y los impactos negativos en una sociedad que subcontrata pueden ser significativos desde la perspectiva de los puestos. El Dr. Mankiw, como consejero de economía del presidente Bush, públicamente promocionó los beneficios de la subcontratación, y fue duramente criticado por esta postura. ¿Cree usted que es bueno o malo para una empresa el subcontratar el trabajo? ¿Cree usted que

existen dilemas éticos para las compañías que subcontratan? Encuentre al menos dos artículos acerca de los impactos de la subcontratación y llévelos para compartirlos en la clase.

3. Usted es un administrador de red y como parte de su trabajo, vigila los correos electrónicos de los empleados. Usted descubre que su jefe está abreviando pasos en un sistema que su compañía desarrolla con el fin de terminar el proyecto más rápido y quedar dentro del presupuesto. Como resultado, hay una falla en el sistema y esta falla ocasionará una caída de la red si más de 20 personas están conectadas al mismo tiempo. El cliente espera aproximadamente 12 personas en la red en un momento dado. Usted está seguro, al parecer igual que su jefe, que el cliente no lo descubrirá hasta mucho después (si acaso) de que el proyecto haya sido aceptado. ¿Qué haría?
4. Un analista de sistemas debe ser capaz de comunicarse técnica y exitosamente con los clientes. Desarrollar un buen sistema requiere una comprensión completa de los requerimientos del usuario. Muchas veces, los usuarios no saben lo que está disponible (tecnológicamente) o incluso lo que les gustaría obtener de un sistema. ¿Cuáles son las características de una buena comunicación?

## Ejercicios de equipo e individuales



1. Reúnanse en grupos de dos personas. La persona uno decidirá acerca de una tarea que él o ella desea completar. Por ejemplo, sacar punta a un lápiz o escribir el nombre del profesor. Debe ser simple y directo. Esa persona debe comunicar en papel con el uso de diagramas sin palabras (verbales ni

escritas) lo que quiere que se haga y dárselo a la persona número dos. La persona número dos debe entonces completar la tarea solicitada.

2. ¿Qué descubrió de este ejercicio? ¿Cuánto tiempo le tomó hasta que la segunda persona entendió lo que la primera persona pedía? ¿Hubo una mala comuni-

cación? Escriba sus pensamientos y observaciones y compártalos con la clase.

3. Ejercicio individual: Imagine una tecnología increíble. El cielo es el límite y cualquier cosa es posible. ¿Cómo

- impacta su vida esta tecnología? ¿Impacta el negocio?
4. Ejercicio individual: Recuerde cuál fue la última vez que alguien le dijo que algo *no podía* hacerse. ¿Qué fue? ¿Les prestó atención? ¿Por qué sí o por qué no?



## Lecturas recomendadas

Ambler, Scott. *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process*. Nueva York: John Wiley & Sons, 2002. Este libro ha dado forma significativa a nuestro pensamiento acerca del proceso de desarrollo de software. Aquellos de ustedes que critican el movimiento de "programación extrema" no deben temer que nuestro entusiasmo por esta lectura recomendada indique un apoyo de la programación extrema. Simplemente nos gusta el juicio que Scott trae al proceso del desarrollo de sistemas y software a través del uso de métodos flexibles dentro del contexto de un proceso iterativo. Haremos referencia a este libro en diversos capítulos.

Ernest, Kallman; John Grillo y James Linderman. *Ethical Decision Making and Information Technology: An Introduction with Cases*, 2a. ed. Burr Ridge, IL, McGraw Hill/Irwin, 1995. Es un texto excelente para enseñar ética en un currículo de MIS. Es una colección de casos de estudio que pueden complementar un curso de análisis y diseño de sistemas.

Simposio y exposición anual de TI de Gartner Group. La unidad de información administrativa de nuestra universidad ha estado suscrita desde hace tiempo al servicio de Gartner Group que reporta las tendencias de la industria, las probabilidades de éxito de tendencias y tecnologías y estrategias sugeridas para una transferencia de tecnología de la información. La investigación de Gartner ha tenido un papel significativo y continuo para ayudarnos a hacer diagramas de los controladores de negocios y de tecnología como se describen en este capítulo. También hemos tenido la fortuna de poder asistir al simposio anual de TI de Gartner. Los informes y simposios

de Gartner Group han influido en cada edición de este libro. Para mayor información acerca de Gartner Group, véase [www.gartner.com](http://www.gartner.com).

Gause, Donald y Gerald Weinberg. *Are Your Lights On? How to Figure Out What the Problem REALLY Is*. Nueva York: Dorset House Publishing, 1990. Sí, este no es un libro reciente, pero tampoco lo son los básicos de solución de problemas. Aquí tenemos un libro breve y de lectura fácil acerca de solución general de problemas. Usted probablemente puede leer el libro completo en una noche y podría mejorar profundamente su potencial de solución de problemas como analista de sistemas (o, en su caso, en cualquier profesión).

Levine, Martin. *Effective Problem Solving*, 2a. ed., Englewood Cliffs, NJ, Prentice Hall, 1994. Éste es otro libro antiguo, pero como afirmamos antes, los métodos de solución de problemas son atemporales. Con sólo 146 páginas, este título puede servir como una excelente referencia profesional.

Weinberg, Gerald, *Rethinking Systems Analysis and Design*. Nueva York, Dorset House Publishing, 1988. No permita que la fecha lo engañe. Es uno de los mejores y más importantes libros acerca de este tema que hayan sido escritos. Este libro puede no enseñar ninguno de los métodos populares de análisis y diseño de nuestros días, pero desafía al lector a ir más allá de esos métodos para considerar algo mucho más importante, el lado de las personas del trabajo de sistemas. Las teorías y conceptos del Dr. Weinberg se presentan en el contexto de docenas de encantadoras fábulas y breves anécdotas. Le agradecemos nuestra fábula de análisis de sistemas favorita de todos los tiempos "The Three Ostriches".



# 2 Requerimientos

En la parte 2 se describen detalladamente los flujos de trabajo del ciclo de vida del software. Para cada flujo de trabajo, se presentan las actividades, las herramientas CASE, métricas y técnicas de pruebas apropiadas para ese flujo de trabajo, así como los retos de cada uno de ellos.

El capítulo 10, “Requerimientos”, examina el flujo de trabajo de los requerimientos. El objetivo de este flujo de trabajo es determinar las necesidades reales del cliente. Se examinan varias técnicas de análisis de requerimientos.

Una vez que los requerimientos se han determinado, el próximo paso es trazar las especificaciones. El enfoque clásico se describe en el capítulo 11, “Análisis clásico”. Se presentan tres enfoques básicos de las especificaciones: informal, semiformal y formal. Se describen instancias de cada enfoque. Se presentan las técnicas descritas con detalle e ilustradas por casos de estudio que incluyen análisis de sistemas estructurados, máquinas de estados finitos, redes de Petri y Z. Se presenta una comparación de las diversas técnicas.

Todas las técnicas de análisis del capítulo 11 son del paradigma clásico. El enfoque orientado a objetos se describe en el capítulo 12, “Análisis orientado a objetos”. Esta técnica orientada a objetos se presenta como una alternativa a las técnicas de análisis clásico del capítulo anterior.

En el capítulo 13, “Diseño”, se compara una variedad de técnicas de diseño, incluyendo aquéllas clásicas como análisis de flujo de información y análisis de transacciones así como diseño orientado a objetos. Se pone especial atención al diseño orientado a objetos incluyendo los casos de estudio. Nuevamente, el énfasis está en la comparación y el contraste.

En el capítulo 14, “Implementación”, se estudian elementos de implementación. Las áreas que se cubren incluyen elementos de implementación, elementos de integración, prácticas correctas de programación y estándares de programación.

El capítulo 15 se titula “Mantenimiento posentrega”. Los tópicos cubiertos en este capítulo incluyen la importancia y el reto del mantenimiento posentrega. La administración del mantenimiento posentrega se estudia meticulosamente.

En el capítulo 16, “Más sobre UML”, se proporciona información adicional acerca del Lenguaje de Modelado Unificado.

Al final de la parte 2, usted deberá tener una clara comprensión de los flujos de trabajo del proceso de software, los retos relacionados con cada flujo de trabajo y cómo cumplir dichos retos.

## Objetivos de aprendizaje

Después de estudiar este capítulo, usted será capaz de

- Realizar el flujo de trabajo de los requerimientos.
- Trazar el modelo de negocio inicial.
- Trazar los requerimientos.
- Construir un prototipo rápido.

Las oportunidades de que un producto se desarrolle a tiempo y dentro del presupuesto son un tanto escasas a menos que los miembros del equipo de desarrollo de software estén de acuerdo con lo que debe realizar el producto software. El primer paso para lograr esta unanimidad es analizar lo más preciso posible la situación actual del cliente. Por ejemplo, es inadecuado decir: "El cliente necesita un sistema de diseño asistido por computadora debido a que él exclama que su sistema de diseño manual es malo". A menos de que el equipo de desarrollo conozca exactamente qué está mal en el sistema manual actual, existe una alta probabilidad de que los aspectos de un sistema computarizado nuevo sean igualmente "malos". De igual forma, si un productor de computadoras personales está considerando la creación de un sistema operativo nuevo, el primer paso es evaluar el sistema operativo actual de la compañía y analizar cuidadosa y exactamente por qué no es satisfactorio. Para tomar un ejemplo extremo, es vital saber si el problema existe sólo en la mente del gerente de ventas, quien culpa al sistema operativo por las ventas bajas, o si los usuarios del sistema operativo están desencantados de su funcionalidad y confiabilidad. Sólo después de que se ha obtenido una imagen clara de la situación presente el equipo puede intentar contestar una pregunta importante: ¿qué debe ser capaz de realizar el producto nuevo? La respuesta a esta pregunta es el objetivo principal del flujo de trabajo de los requerimientos.

### 10.1 Cómo determinar lo que el cliente necesita

Un concepto que erróneamente se tiene es que, durante el flujo de trabajo de los requerimientos, los desarrolladores deben determinar qué software es el que el cliente *quiere*. Por el contrario, el objetivo real del flujo de trabajo de los requerimientos es determinar qué software es el que el cliente *necesita*. Un problema es que muchos clientes no saben lo que necesitan. Aún más, incluso un cliente que tiene una buena idea de lo que necesita podría tener dificultades en transmitir correctamente estas ideas a los desarrolladores, puesto que la mayoría de los clientes están menos enterados sobre computadoras que los miembros del equipo de desarrollo. (Para adentrarse más en este punto, véase *Por si desea saber*, cuadro 10.1.)

Otro problema es que el cliente quizás no aprecie qué es lo que está sucediendo en su organización. Por ejemplo, es inútil que el cliente solicite un producto software más rápido cuando la verdadera razón por la cual el actual software tiene un tiempo tan grande de respuesta es porque la base de datos está mal diseñada. Lo que se necesita es reorganizar y mejorar la forma en que la información se almacena en el producto software actual. Un producto software nuevo sería igualmente lento. O, si el cliente opera una cadena de tiendas departamentales sin utilidades, podría solicitar un sistema de administración de información financiera que refleje elementos como ventas, salarios, cuentas por pagar y cuentas por cobrar. Tal sistema de información será de muy poca utilidad si la razón verdadera de sus pérdidas es la merma (robo en tienda y empleados que roban). Si ése es el caso, entonces se requiere un sistema de control de inventarios más que un sistema de administración de información financiera.

A primera vista, es sencillo determinar lo que el cliente necesita: los miembros de los equipos de desarrollo simplemente le preguntan. Sin embargo, existen dos razones de porqué este enfoque directo comúnmente no funciona muy bien.

Primero, tal como se estableció, el cliente pudiera no apreciar qué sucede en su propia organización. Pero la razón principal de porqué el cliente continuamente solicita el producto software erróneo se debe a que el software es complejo. Para un analista de sistemas resulta bastante difícil visualizar un producto software y su funcionalidad; el problema es aún peor para el cliente, quien regularmente es inexperto en ingeniería de software.

## CUADRO 10.1

S. I. Hayakawa (1906-1992), senador estadounidense de California, una vez declaró a un grupo de periodistas: "Yo sé que ustedes creen que entendieron lo que piensan que les dije, pero no estoy seguro de que ustedes se hayan dado cuenta de que lo que escucharon no es a lo que yo me refería". Esta excusa aplica de igual manera al punto del análisis de requerimientos. Los analistas de sistemas escucharon las peticiones de su cliente, pero lo que ellos escucharon no es lo que el cliente debería estar diciendo.

Esa cita se ha atribuido erróneamente al candidato a la presidencia de los Estados Unidos George Romney (1907-1995), quien alguna vez anunció en una conferencia de prensa: "Yo no dije que yo no lo dije. Yo dije que yo no dije que yo lo dije. Quiero que esto quede muy claro". La "clarificación" de Romney subraya otro reto del análisis de requerimientos; es fácil confundir lo que el cliente dice.

Por si desea saber

Sin la ayuda de un equipo de desarrollo de software talentoso, el cliente pudiera ser una fuente de información pobre de acuerdo con lo que se necesita desarrollar. Por otro lado, a menos de que exista una comunicación cara a cara con el cliente, no hay manera de encontrar lo que realmente se necesita.

El intento clásico por resolver este reto se describe en la sección 10.11. El enfoque orientado a objetos consiste en obtener información inicial por parte del cliente y usuarios futuros del producto y utilizarla como la entrada del flujo de trabajo de los requerimientos del Proceso Unificado [Jacobson, Booch y Rumbaugh, 1999]. Esto se describe en la siguiente sección.

## 10.2 Repaso del flujo de trabajo de los requerimientos

El primer paso en el **flujo de trabajo de los requerimientos** es comprender el **dominio de la aplicación** (**dominio**, para abreviar), esto es, el ambiente específico en el cual operaría el producto deseado. El dominio pudiera ser bancos, exploración espacial, industria automotriz o telemetría. Una vez que los miembros del equipo de desarrollo entiendan el dominio con la debida profundidad, pueden construir un modelo de negocio, esto es, utilizar diagramas UML para describir los procesos de negocio del cliente. El modelo de negocio se utiliza para determinar cuáles son los requerimientos iniciales del cliente. Entonces se aplica la iteración.

En otras palabras, el punto de partida es la comprensión inicial del dominio. Esta información se utiliza para construir el modelo de negocio inicial. Éste se utiliza para trazar un conjunto inicial de los requerimientos del cliente. Entonces, a la luz de lo que se ha aprendido acerca de dichos requerimientos, se obtiene una comprensión más profunda del dominio; y este conocimiento se utiliza a su vez para detallar el modelo de negocio y por lo mismo los requerimientos del cliente. Esta iteración continúa hasta que el equipo está satisfecho con el conjunto de requerimientos. En este punto, la iteración se detiene.

El proceso de descubrir los requerimientos del cliente se denomina **extracción de requerimientos** (o **captura de requerimientos**). Una vez que se traza el conjunto inicial de requerimientos, el proceso de detallarlos y extenderlos se denomina **análisis de requerimientos**.

Ahora examinamos estos pasos en detalle.

## 10.3 Comprender el dominio

Para extraer las necesidades del cliente, los miembros del equipo de requerimientos deben familiarizarse con el dominio de la aplicación, esto es, el área general en la cual se usará el producto deseado.

Por ejemplo, no es fácil realizar preguntas significativas a un banquero o a un neurocirujano sin primero familiarizarse con la banca o la neurocirugía. De ahí que una tarea inicial de cada miembro del equipo de análisis de requerimientos es familiarizarse con el dominio de la aplicación, a menos que él o ella ya tengan experiencia en esa área. Es muy importante usar la terminología correcta cuando se comunique con el cliente y usuarios potenciales del software deseado. Después de todo, es difícil que lo tomen a uno como una persona que trabaja en un dominio específico a menos que el entrevistador use la nomenclatura apropiada para ese dominio. Más importante, el uso de una palabra inapropiada pudiera ocasionar una confusión, que eventualmente resultaría en una entrega de producto fallido. El mismo problema pudiera surgir si los miembros del equipo de requerimientos no comprenden las sutilezas de la terminología del dominio. Por ejemplo, para una persona común palabras como *abrazadera*, *barra*, *trabe* y *poste* pudieran ser sinónimos, pero para un ingeniero civil son términos distintos. Si un desarrollador no aprecia que un ingeniero civil utiliza estos cuatro términos en una forma precisa y si el ingeniero civil asume que el desarrollador está familiarizado con las diferencias de estos términos, el desarrollador pudiera tratar los cuatro términos como equivalentes; el software de diseño de puentes asistido por computadora resultante pudiera contener fallas que ocasionarían un colapso del puente. Los profesionales de computación esperan que la salida de cualquier programa será escrutinada cuidadosamente por un humano antes de que las decisiones se hagan basándose en ese programa, pero la fe popular y creciente en computadoras supone que es tonto que se haga esa revisión. Por lo mismo, no es una idea descabellada que la confusión de términos pudiera ocasionar que los desarrolladores de software sean demandados por negligencia.

Una forma de tratar el problema con la terminología es construir un **glosario**, una lista de palabras técnicas utilizadas en el dominio, junto con sus significados. Las entradas iniciales se insertan en el glosario mientras que los miembros del equipo se ocupan en aprender lo más que puedan acerca del dominio de la aplicación. Después, el glosario se actualiza cuando cualesquiera de los miembros del equipo de requerimientos encuentre terminología nueva. De vez en cuando, el glosario puede imprimirse y distribuirse a los miembros del equipo o descargarse a un dispositivo portátil (PDA) (tal como una Palm Pilot). Dicho glosario no sólo reduce la confusión entre el cliente y los analistas de sistemas, también es útil para disminuir malentendidos entre los miembros del equipo de desarrollo.

Una vez que el equipo de requerimientos se ha familiarizado con el dominio, el siguiente paso es construir el modelo de negocio.

## 10.4 El modelo de negocio

Un **modelo de negocio** es una descripción de los procesos de negocio de una organización. Por ejemplo, algunos de los procesos de negocio de un banco incluyen aceptar depósitos de los clientes, prestar dinero a los clientes y hacer inversiones.

La razón para construir un modelo de negocio se debe primero a que éste proporciona una comprensión del negocio del cliente como un todo. Con este conocimiento, los desarrolladores pueden aconsejar al cliente acerca de qué partes del negocio sistematizar. Asimismo, si la tarea es extender un producto software existente, los desarrolladores tienen que entender el negocio existente como un todo para determinar cómo incorporar la extensión y aprender qué partes, si es que alguna, del producto existente necesitan ser modificadas para agregar el nuevo componente.

Para construir un modelo de negocio, un analista de sistemas necesita obtener una comprensión detallada de los distintos procesos de negocio. Estos procesos ahora se *refinan*, esto es, se analizan con mayor detalle. Se pueden utilizar diferentes técnicas para obtener la información necesaria para construir el modelo de negocio, principalmente entrevistas.

### > 10.4.1 Cómo entrevistar

Los miembros del equipo de requerimientos se reúnen con los de la organización cliente hasta que se convencen de que han extraído toda la información relevante del cliente y de usuarios futuros del producto software deseado.

Existen dos tipos básicos de preguntas. Una pregunta cerrada requiere una respuesta específica. Por ejemplo, al cliente se le podría preguntar cuánto personal de ventas emplea la compañía o qué

tan rápido se requiere una respuesta. Se realizan preguntas abiertas para promover que la persona entrevistada hable. Por ejemplo, preguntar al cliente, “¿Por qué su producto de software actual no es satisfactorio?” pudiera explicar muchos aspectos del enfoque que el cliente tiene del negocio. Algunos de estos aspectos pudieran no salir a la luz si la pregunta fuera cerrada.

De igual forma, existen dos tipos básicos de entrevista, estructurada y no estructurada. En una **entrevista estructurada** se realizan preguntas preplaneadas, casi siempre cerradas. En una **entrevista no estructurada**, el entrevistador pudiera comenzar con una o dos preguntas cerradas, pero se plantearían subsecuentes preguntas en respuesta a las respuestas que él o ella reciban de la persona entrevistada. Se intentaría que muchas de estas preguntas subsecuentes fueran abiertas para proporcionar al entrevistador un amplio rango de información.

Al mismo tiempo, no es una buena idea que el entrevistador haga una entrevista demasiado no estructurada. Decirle al cliente, “hábleme de su negocio” no produciría un conocimiento relevante. En otras palabras, las preguntas deberán plantearse de tal forma que busquen que la persona entrevistada dé respuestas amplias pero siempre dentro del contexto de la información específicamente útil para el entrevistador.

Llevar a cabo una buena entrevista no siempre es fácil. Por principio, el entrevistador debe estar completamente familiarizado con el dominio de la aplicación. Segundo, no tiene sentido entrevistar a un miembro de la organización cliente, si el entrevistador ya tiene su propio juicio acerca de las necesidades del cliente. No importa lo que se le haya dicho previamente al entrevistador o lo que él o ella haya averiguado por algún otro medio, el entrevistador debe enfocar cada entrevista con la intención de escuchar cuidadosamente lo que la persona entrevistada tiene que decir, a la vez eliminando estrictamente cualquier noción preconcebida referente a la compañía cliente o las necesidades de los clientes y de los usuarios potenciales del producto que se desea desarrollar.

Después de concluir la entrevista, el entrevistador debe preparar un informe escrito en el que incorpore los resultados de la entrevista. Se aconseja siempre proporcionar una copia del informe a la persona que se entrevistó; él o ella pudieran querer aclarar ciertos argumentos o agregar elementos sobrepasados.

#### > 10.4.2 Otras técnicas

La entrevista es la técnica principal para obtener información sobre el modelo de negocio. Esta sección describe algunas otras técnicas que pudieran usarse junto con las entrevistas.

Una forma de adquirir conocimiento acerca de las actividades de la organización cliente es enviar un **uestionario** a los miembros relevantes de la organización cliente. Esta técnica es útil cuando las opiniones de cientos de individuos necesitan ser determinadas. Aún más, una respuesta escrita pensada con sumo cuidado por parte de un empleado de la organización cliente puede ser más acertada que una respuesta verbal inmediata a una pregunta que plantee el entrevistador. Sin embargo, una entrevista no estructurada realizada por un entrevistador metódico que escucha con sumo cuidado y plantea preguntas que amplían las respuestas iniciales, usualmente genera mejor información que un cuestionario conscientemente contestado. Debido a que los cuestionarios están preplaneados, no existe ninguna forma de plantear una pregunta en consecuencia a una respuesta.

Una forma diferente de extraer requerimientos es examinar las distintas **formas** que utiliza el negocio. Por ejemplo, una forma de trabajos impresos puede reflejar el número de prensa, el tamaño del rodillo de papel, humedad, temperatura de la tinta, tensión del papel, etc. Los distintos aspectos de esta forma dan información sobre el flujo de los trabajos de impresión y la importancia relativa de los pasos en el proceso de impresión. Otros documentos, como los procedimientos operativos y las descripciones de trabajo también pueden ser herramientas poderosas para conocer exactamente qué se hace y cómo. Si un producto de software ya se usa, también se deben estudiar cuidadosamente los manuales de usuario. Un conjunto de distintos tipos de datos sobre la forma en que el cliente actualmente hace negocios puede ser de gran ayuda para determinar las necesidades del cliente. Es por esto que un buen profesional de software estudia cuidadosamente la documentación del cliente, tratándola como una valiosa fuente de información sobre las necesidades del cliente.

Otra forma de obtener tal información es por **observación directa** de los usuarios; esto es, que miembros del equipo de requerimientos observen y registren las acciones de los empleados mientras ellos desempeñan sus labores. Una versión moderna de esta técnica es instalar **cámaras de video** dentro del espacio de trabajo para registrar (con el permiso previamente escrito de aquellos que están

siendo observados) exactamente lo que se hace. Un obstáculo de esta técnica es que analizar las cintas puede tomar mucho tiempo. En general, uno o más miembros del equipo de requerimientos tiene que gastar una hora revisando la cinta por cada hora que las cámaras registraron. Este tiempo se añade al que se necesita para valuar lo observado. Más seriamente, se sabe que esta técnica puede volverse en contra de quien la usa, ya que los empleados pudieran pensar que la instalación de las cámaras es una invasión a su privacidad. Es importante que el equipo de requerimientos obtenga la completa cooperación de todos los empleados; puede ser muy difícil conseguir la información necesaria si las personas se sienten amenazadas u hostigadas. Los posibles riesgos se deberán considerar con sumo cuidado antes de introducir cámaras o, de hecho, tomar cualquier otra acción que pueda molestar o incluso enojar a los empleados.

### > 10.4.3 Casos de uso

Como se estableció en la sección 3.2, un **modelo** es un conjunto de diagramas UML que representan uno o más aspectos del producto de software a ser desarrollado (hay que recordar que la *ML* en UML significa “lenguaje de modelado”). Uno de los principales diagramas UML utilizados en el modelado de negocio es el caso de uso.

Un **caso de uso** modela la interacción entre el producto de software en sí y los usuarios del producto de software (**actores**). Por ejemplo, la figura 10.1 ilustra un caso de uso de un producto de software de un banco. Ahí hay dos actores, representados por las figuras de hombrecitos en UML, el **Cliente** y el **Cajero**. La etiqueta dentro del óvalo describe la actividad del negocio representada por el caso de uso, en este caso Retiro de Dinero.



**FIGURA 10.1** El caso de uso Retiro de Dinero del producto de software del banco.

Otra forma de ver un caso de uso es que éste muestra la interacción entre el producto de software y el ambiente en el cual opera dicho producto. Esto es, un actor es miembro del mundo externo del producto de software, mientras que el rectángulo en el caso de uso representa el producto de software en sí.

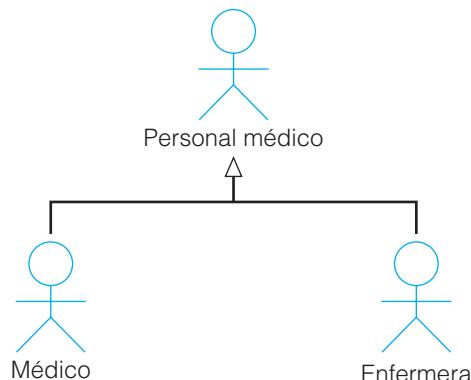
Casi siempre es fácil identificar un actor. Con frecuencia un actor es un usuario del producto de software. En el caso de producto de software del banco, los usuarios del producto de software son los clientes del banco y el personal del banco, incluyendo cajeros, gerentes, etc. En general, un actor desempeña una función respecto al producto de software. Esta función puede ser como usuario del producto de software. Sin embargo, el iniciador del caso de uso o alguien que desempeñe una parte importante en un caso de uso también está desempeñando una función y por esto se le considera un actor, sin tomar en cuenta si esa persona también es un usuario del producto de software. Un ejemplo de esto se proporciona en la sección 10.7.

Un usuario del sistema puede desempeñar más de una función. Por ejemplo, un cliente del banco puede ser un **Prestatario** (cuando él o ella toman un préstamo) o un **Prestamista** (cuando él o ella depositan dinero en el banco; un banco genera muchas de sus ganancias invirtiendo el dinero que depositan sus clientes). De igual forma, un actor puede participar en múltiples casos de uso. Por ejemplo, un **Prestatario** puede ser un actor en el caso de uso **Prestar Dinero**, en el caso de uso **Pagar Interés sobre el Préstamo** y en el caso de uso **Repagar el Préstamo Principal**. También el actor **Prestatario** pudiera representar a muchos de los clientes del banco.

Un actor no necesariamente tiene que ser un humano. Habrá que recordar que un actor es un usuario de un producto de software, y en muchos casos, otro producto de software puede ser un usuario. Por ejemplo, un sistema de información de comercio electrónico que permita a los compradores pagar con tarjetas de crédito tiene que interactuar con el sistema de información con la

compañía de tarjetas de crédito. Esto es, el sistema de información de la compañía de tarjetas de crédito es un actor desde el punto de vista del sistema de información de la compañía de comercio electrónico. De igual forma el sistema de información de comercio electrónico es un actor desde el punto de vista del sistema de información de la compañía de tarjetas de crédito.

Como se estableció previamente, la identificación de los actores es sencilla. Por lo general, la única dificultad que aparece en esta parte del paradigma es que un profesional de software experimentado algunas veces identifica actores que se traslanan. Por ejemplo, en un producto de software de un hospital, tener un caso de uso con el actor **Enfermera** y un caso de uso diferente con el actor **Personal médico** no es una buena idea, ya que todas las enfermeras son personal médico, pero algunos del personal médico (tal como médicos) no son enfermeras. Sería mejor tener a los actores **Médico** y **Enfermera**. Alternativamente, el actor **Personal médico** puede definirse con dos especializaciones **Médico** y **Enfermera**. Esto se ilustra en la figura 10.2. En la sección 7.7 se señaló que la herencia es un caso especial de la generalización. La generalización se aplicó a las clases en la sección 7.7. La figura 10.2 muestra cómo se aplica la generalización también a los actores.



**FIGURA 10.2** Generalización del personal médico.

## 10.5 Requerimientos iniciales

Para determinar los requerimientos del cliente, se trazan los requerimientos iniciales basados en el modelo de negocio inicial. Después, según se vaya comprendiendo el dominio y el modelo de negocio se detalle con base en pláticas subsecuentes con el cliente, se detallan los requerimientos.

Los requerimientos son dinámicos. Esto es, existen cambios frecuentes no sólo en los requerimientos en sí, sino también en las actitudes del equipo de desarrollo, cliente y usuarios futuros hacia cada requerimiento. Por ejemplo, un requerimiento particular en un principio pudiera parecerle al equipo de desarrollo como opcional. Después de un análisis posterior, pudiera parecer que el requerimiento es de importancia crítica. Sin embargo, después de comentarlo con el cliente, se elimina el requerimiento. Una manera apropiada de manejar estos cambios frecuentes es mantener una lista de posibles requerimientos, junto con los casos de uso de los requerimientos en los que ya se han puesto de acuerdo los miembros del equipo de desarrollo y que el cliente ha aprobado.

Es importante tener en mente que el paradigma orientado a objetos es iterativo y que el glosario, el modelo de negocio o los requerimientos por esto mismo pudieran tener que modificarse en cualquier momento. En particular, adiciones a la lista de requerimientos, modificaciones a los elementos que ya están en la lista y eliminación de elementos de dicha lista se pudieran generar por una amplia variedad de eventos, manteniendo un rango que va desde una observación casual que un usuario haga hasta una sugerencia que el cliente realice en una junta formal de analistas de sistemas del equipo de desarrollo. Cualquier cambio de éstos pudiera ocasionar cambios al modelo de negocio.

Los requerimientos caen en dos categorías, funcionales y no funcionales. Un **requerimiento funcional** especifica una acción que deberá ser capaz de desempeñar el producto deseado. Los reque-

rimientos funcionales regularmente se expresan en términos de entradas y salidas: dada una entrada específica, el requerimiento funcional estipula cuál debe ser la salida. A la inversa, un **requerimiento no funcional** especifica propiedades del producto deseado en sí, como **restricciones de la plataforma** (“el producto de software debe ejecutarse en Linux”), **tiempos de respuesta** (“en promedio, las consultas de Tipo 3B deberán responderse dentro de 2.5 segundos”), o **confiabilidad** (“el producto de software debe ejecutarse en 99.95% del tiempo”).

Los requerimientos funcionales se manejan mientras los flujos de trabajo de los requerimientos del análisis están siendo desarrollados, mientras que algunos requerimientos no funcionales pudieran tener que esperar al flujo de trabajo del diseño. La razón se debe a que, para ser capaz de manipular ciertos requerimientos no funcionales, se pudiera necesitar el conocimiento detallado del producto de software deseado y este conocimiento por lo regular no está disponible hasta que los flujos de trabajo de los requerimientos y del análisis se han completado (véanse problemas 10.2 y 10.3). Sin embargo, en donde sea posible, los requerimientos no funcionales también se deberán manipular durante los flujos de trabajo de los requerimientos y el análisis.

El flujo de trabajo de los requerimientos se ilustra ahora mediante un caso de estudio.

## 10.6 Comprensión inicial del dominio: el caso de estudio Osbert Oglesby

Osbert Oglesby, el renombrado distribuidor de arte, necesita un producto de software que lo ayude a comprar y vender pinturas. Osbert se especializa en comprar y vender pinturas del Impresionismo francés. Sin embargo, él comprará prácticamente cualquier pintura si piensa que puede obtener una utilidad cuando la venda en su galería, Les Objets d’Orient.

Es necesario entrevistar a Osbert para obtener información detallada acerca de cualquier aspecto de su negocio. Pero antes de realizar esto, se debe adquirir conocimiento en el dominio, esto es, información acerca de pinturas y del negocio del arte en general. Esta información fundamental permite al equipo de requerimientos comprender cualesquiera términos técnicos utilizados por Osbert cuando se reúnan con él. Además, si conocen algo acerca del negocio del arte en general antes de reunirse con él, ellos pueden identificar cualesquiera aspectos aparentemente inusuales sobre la forma en que Osbert hace negocios. Entonces ellos pudieran preguntarle acerca de esos aspectos. En algunas instancias, ellos podrán sugerir que, haciendo lo mismo que la mayoría de sus competidores están realizando, él podrá incrementar las utilidades de su negocio. En otros casos, los aspectos únicos de las prácticas de negocio de Osbert pudieran darle una ventaja competitiva. Es importante incorporar todos estos casos dentro del producto de software que se le desarrollará.

Para aprender sobre pinturas y del negocio del arte en general, el equipo de requerimientos visita los sitios Web de dos de los principales museos de arte y de dos distribuidores de arte. El equipo descubre que existen distintas maneras de clasificar las pinturas. Una es considerar el **medio**, esto es, el material con el que se pintó el trabajo de arte. El medio más popular es la pintura basada en aceite (**óleo**), pero algunas pinturas más pequeñas se elaboran con pintura basada en agua (**acuarela**). En ocasiones se utilizan **otros medios**, incluyendo lápiz, pastel y crayón, pero con menor frecuencia. Ellos colocan esta información en su glosario, mostrado en la figura 10.3.

Una segunda manera de clasificar una pintura es por el **tema**. El tema más popular es una persona (**retrato**). Otros temas comunes incluyen la naturaleza (**paisaje**) y un objeto inanimado como un jarrón de flores, un tazón de fruta, etc. (**naturaleza muerta**). Esta información se agrega al glosario.

Una tercera clasificación es la **calidad** de la pintura. Una pintura de indudable excelencia se denomina una **obra maestra**. Un **trabajo maestro** es una pintura inferior de un artista que previa o subsecuentemente ha pintado una obra maestra. Sin embargo, la basta mayoría de las pinturas no son ni obras maestras ni trabajos maestros (**otras pinturas**). Esta información también se coloca en el glosario, tal como se muestra en la figura 10.3.

El modelo de negocio inicial del caso de estudio Osbert Oglesby se construye ahora.

**Paisaje:** una pintura de una escena en la naturaleza

**Obra maestra:** una pintura de indudable excelencia

**Trabajo maestro:** una pintura inferior de un artista que previamente o subsecuentemente ha pintado una obra maestra

**Medio:** un criterio de clasificación; el material con el que se pinta un trabajo de arte; véase también óleo, acuarela

**Óleo:** un medio; abreviación de “pintura basada en aceite”

**Otras pinturas:** una pintura que no es ni una obra maestra ni un trabajo maestro

**Retrato:** una pintura de una o más personas

**Calidad:** un criterio de clasificación; una pintura se clasifica como una obra maestra, trabajo maestro u otra pintura, dependiendo de su calidad

**Naturaleza muerta:** una pintura de objetos inanimados

**Tema:** un criterio de clasificación; los temas incluyen paisaje, retrato y naturaleza muerta

**Acuarela:** un medio; abreviación de “pintura basada en agua”

**FIGURA 10.3** El glosario inicial del caso de estudio Osbert Oglesby.

## 10.7 Modelo de negocio inicial: el caso de estudio Osbert Oglesby

El negocio de Osbert Oglesby ha sido exitoso por muchos años. Últimamente, sin embargo, Osbert ha perdido dinero. Un consultor gerencial analizó los registros del negocio y concluyó que Osbert ha estado pagando de más por las pinturas. El consultor notificó a Osbert que adquiriera un producto de software que se ejecute en una computadora *laptop*, que él pueda utilizar para determinar el precio máximo a pagar por una pintura. Después Osbert pudiera utilizar el producto de software en su galería o llevarse la computadora cuando vea una pintura en la casa u oficina de un comprador. (Osbert se refiere a sus compradores como sus clientes. En este libro, sin embargo, la palabra *cliente* se utiliza para la persona que quiere que se desarrolle un producto de software, en este caso, Osbert.)

El consultor gerencial derivó un algoritmo que el producto de software deberá usar para determinar el precio máximo que se debe pagar por una pintura. En este punto de la metodología, es muy temprano para revisar la fórmula. Después de todo, la Ley de Miller aplica para todos los humanos. Estamos limitados en el número de pedazos de información que podemos manipular en cualquier tiempo. Por lo pronto, todo lo que tenemos que saber son los procesos de negocio de Osbert; los detalles de la fórmula pueden esperar hasta una siguiente iteración.

Osbert quiere que el producto de software calcule el precio máximo a ofrecer cuando se compre una pintura. A diferencia de la mayoría de los distribuidores, Osbert no cree en el regateo. Cuando él quiere comprar una pintura, la cantidad que ofrece es su precio final. De igual forma, cuando vende una pintura, él no negocia; el precio que da es el precio final.

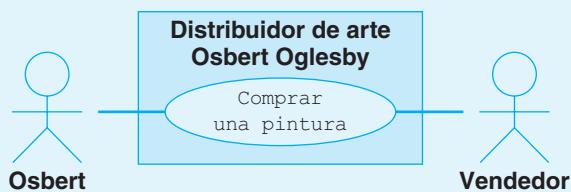
Además, Osbert quisiera que el producto de software detectara las nuevas tendencias en el mercado del arte tan pronto como sea posible. Él se interesa sobre todo en determinar cuándo se están pagando de manera consistente precios más altos que lo esperado por un trabajo particular de un artista, de esta forma él puede comprar pinturas de ese artista antes de que otros se den cuenta de la tendencia.

Para avisar cuando el precio de venta de una pintura sea mayor a lo que Osbert esperaba cuando compró la pintura, el producto de software necesita mantener un registro de

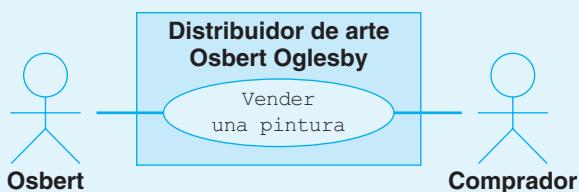
todas sus compras y todas sus ventas. En tanto que esa información esté disponible en su computadora, Osbert podría querer utilizarla para obtener información adicional de su negocio. En particular, hoy en día Osbert genera dos informes a mano, uno que muestra todas las compras de pinturas durante el año pasado y otro que muestra todas las ventas durante el año pasado. Por un pequeño costo adicional, el producto de software puede generar estos dos informes según se requiera.

En otras palabras, lo que Osbert *quiere* es un producto de software que calcule el precio más alto que él deberá pagar por una pintura y que también detecte las nuevas tendencias en el arte. Lo que él *necesita* es un producto de software que, además, proporcione informes de compras y ventas. Si los desarrolladores no determinan este hecho al principio, tendrán que incorporar funcionalidad adicional después de haber entregado el producto de software. Si se toma en cuenta la figura 1.6, el costo del producto de software según se modifique será considerablemente mayor al costo pactado del sistema original. Además, Osbert no estará contento con los desarrolladores. En el mejor de los casos, los considerará incompetentes; en el peor de los casos, verá el cargo adicional que tendrá que pagar como una evidencia de su deshonestidad. De manera alterna, los desarrolladores podrían acordar incorporar los informes libres de cargo. Sin embargo, esto significa que cargarían con una pérdida significativa en el producto de software de Osbert. Por fortuna, los analistas de sistemas obviaron el problema determinando las necesidades reales de Osbert.

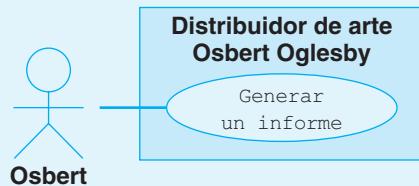
Ahora que se ha obtenido la información inicial, se puede construir el modelo de negocio inicial. Osbert tiene tres actividades de negocio: él compra pinturas, vende pinturas y genera informes, todo manualmente. Los casos de uso correspondientes a Comprar una pintura, Vender una pintura y Generar un informe se muestran en las figuras 10.4, 10.5 y 10.6. La única persona que actualmente usa el sistema (manual) es Osbert, así que él es el actor en los tres casos de uso; sin embargo, como se estableció en la sección 10.4.3, alguien que inicia un caso de uso es también un actor, como también alguien que desempeña una parte importante. El cliente pudiera iniciar el caso de uso Comprar una pintura o Vender una pintura y seguramente desempeñaría una parte importante en ambos casos de uso proporcionando información introducida en el producto de software de Osbert. El cliente es entonces un actor en ambos casos de uso. Se espera que otros actores se agreguen en una iteración posterior. Para abreviar, los tres casos de uso se combinan en un **diagrama de caso de uso** de la figura 10.7; un diagrama de caso de uso es un diagrama UML que refleja dos o más casos de uso.



**FIGURA 10.4** El caso de uso Comprar una pintura del modelo de negocio inicial del caso de estudio Osbert Oglesby.



**FIGURA 10.5** El caso de uso Vender una pintura del modelo de negocio inicial del caso de estudio Osbert Oglesby.



**FIGURA 10.6** El caso de uso Generar un informe del modelo de negocio inicial del caso de estudio Osbert Oglesby.



**FIGURA 10.7** El diagrama del caso de uso del modelo de negocio inicial del caso de estudio Osbert Oglesby.

#### Descripción breve

El caso de uso Comprar una pintura permite a Osbert Oglesby comprar una pintura.

#### Descripción paso a paso

No aplica en este estado inicial.

**FIGURA 10.8** La descripción del caso de uso Comprar una pintura del modelo de negocio inicial del caso de estudio Osbert Oglesby.

#### Descripción breve

El caso de uso Vender una pintura permite a Osbert Oglesby vender una pintura.

#### Descripción paso a paso

No aplica en este estado inicial.

**FIGURA 10.9** La descripción del caso de uso Vender una pintura del modelo de negocio inicial del caso de estudio Osbert Oglesby.

**FIGURA 10.10** La descripción del caso de uso Generar un informe del modelo de negocio inicial del caso de estudio Osbert Oglesby.

Después los casos de uso necesitan ser anotados, de esta forma se agregan al inicio las **descripciones del caso de uso** de las figuras 10.8 a 10.10. Ahora se tienen que trazar los requerimientos iniciales.

## 10.8 Requerimientos iniciales: el caso de estudio Osbert Oglesby

El modelo de negocio inicial (los tres casos de uso de la figura 10.7) muestra cómo se conduce el negocio de Osbert actualmente. El próximo paso es decidir cuáles de estos casos de uso son también requerimientos que se deben construir del producto de software. Después, los requerimientos iniciales resultantes se detallan agregando, modificando y eliminando requerimientos, hasta que Osbert y el equipo de requerimientos estén satisfechos con el conjunto de requerimientos a implementarse.

Los tres casos de uso del modelo de negocio inicial (figura 10.7) –Comprar una pintura, Vender una pintura y Generar un informe– serán requerimientos iniciales. Además, los tres procesos serán automatizados en el producto de software deseado, de esta forma las descripciones iniciales mostradas en las figuras 10.8 a 10.10 son reemplazadas por las descripciones de las figuras 10.11 a 10.13.

La imprecisión de estas descripciones es una consecuencia de la naturaleza iterativa del paradigma orientado a objetos. Por ejemplo, considere la figura 10.11. El elemento uno se refiere a **detalles de la pintura que [Osbert] considera al comprar**. Los miembros del equipo de requerimientos aún no conocen cuáles son estos detalles y definitivamente no desean conocerlos en esta etapa. Un principio básico es postergar los detalles lo más posible. La razón para esto es que, cuando los cambios inevitables tengan que hacerse en la próxima iteración, el número de elementos que tengan que modificarse sea lo más pequeño posible.

### Descripción breve

El caso de uso Comprar una pintura permite que Osbert Oglesby compre una pintura.

### Descripción paso a paso

1. Osbert introduce los detalles de la pintura que considera comprar.
2. El producto de software responde con el precio de compra máximo que deberá ofrecer.

3. Si el comprador acepta la oferta de Osbert para comprar la pintura, Osbert ingresa más detalles.

**Nota:** Los detalles de algoritmo para determinar el precio máximo se obtendrán más tarde.

**FIGURA 10.11** La descripción del caso de uso Comprar una pintura de los requerimientos iniciales del caso de estudio Osbert Oglesby.

#### Descripción breve

El caso de uso Vender una pintura permite que Osbert Oglesby venda una pintura.

#### Descripción paso a paso

1. Osbert introduce detalles de la pintura que vendió.

**FIGURA 10.12** La descripción del caso de uso Vender una pintura de los requerimientos iniciales del caso de estudio Osbert Oglesby.

#### Descripción breve

El caso de uso Generar un informe le permite a Osbert Oglesby obtener información sobre pinturas que compró o vendió el año pasado o detectar nuevas tendencias en el mercado del arte.

#### Descripción paso a paso

1. Osbert solicita un informe del tipo que necesita. El informe se imprime.

**FIGURA 10.13** La descripción del caso de uso Generar un informe de los requerimientos iniciales del caso de estudio Osbert Oglesby.

## CUADRO 10.2

Los requerimientos para el caso de estudio de Osbert Oglesby establecen que la incertidumbre en cualquiera de los cuatro elementos de información, incluyendo el título de la pintura, se deben indicar con dicha información seguida de ?. Sin embargo, el título de una pintura puede incluir un signo de interrogación. Por ejemplo, mientras vivía en Tahití en 1897, Paul Gauguin pintó lo que probablemente sea su mejor trabajo, titulado *D'où venons-nous? Que sommes-nous? Où allons-nous?* (¿De dónde venimos? ¿Qué somos? ¿A dónde vamos?).

Si la implementación del caso de estudio Osbert Oglesby satisface sus requerimientos, Osbert no podrá ingresar el título de la obra maestra de Gauguin correctamente, en el muy poco probable caso que el Museo de Artes Finas, Boston, le venda la pintura (pero vea el problema 15.15).

Por si desea saber

## 10.9 Continuación del flujo de trabajo de los requerimientos: el caso de estudio Osbert Oglesby

Considere las versiones actuales de cada una de las descripciones de los casos de uso (figuras 10.11 a 10.13). Queda claro que se necesita obtener más detalles para la siguiente iteración.

Primero considere los casos de uso Comprar una pintura y Vender una pintura. Para detallar la descripción, se tienen que determinar los atributos necesarios para registrar una pintura cuando se compra y cuando se vende. Examinando la documentación relevante, sobre todo los registros manuales y actuales de Osbert, el equipo de requerimientos aprende que Osbert necesita registrar la siguiente información por cada pintura que compra:

Descripción de la pintura:

Nombre del artista (hasta 20 caracteres, seguidos de ? si existe incertidumbre)  
Apellido del artista (hasta 20 caracteres, seguidos de ? si existe incertidumbre)  
Título del trabajo (hasta 40 caracteres, seguidos de ? si existe incertidumbre)  
Fecha del trabajo (aaaa, seguido de ? si existe incertidumbre)  
Clasificación (obra maestra, trabajo maestro, otra pintura)  
Altura (cm)  
Ancho (cm)  
Medio (óleo, acuarela, otro medio)  
Tema (retrato, naturaleza muerta, paisaje, otro tema)

Fecha de compra (mm/dd/aaaa)

Nombre del vendedor (hasta 30 caracteres)

Dirección del vendedor (hasta 40 caracteres)

Precio máximo de compra determinado por el algoritmo (hasta \$99,999,999)

Precio de compra real (hasta \$99,999,999)

Precio de venta deseado (2.15 veces el precio de compra)

Después de haber vendido la pintura, Osbert debe registrar lo siguiente:

Fecha de venta (mm/dd/aaaa)

Nombre del comprador (hasta 30 caracteres)

Dirección del comprador (hasta 40 caracteres)

Precio de venta real (hasta \$99,999,999)

(Existe un posible problema con estos requerimientos, tal como se describe en Por si desea saber, cuadro 10.2.)

Para calcular el "Precio de compra máximo determinado por el algoritmo", el equipo de requerimientos ahora revisa el informe del consultor gerencial. El algoritmo que desarrolla el consultor es el siguiente:

Clasificar la pintura como una obra maestra, un trabajo maestro u otra pintura.

- **Obra maestra.** Buscar en los registros de subastas alrededor del mundo durante los últimos 25 años el trabajo más parecido del mismo artista, ignorando cualquier signo de interrogación al principio o al final del nombre del artista o en el título o la fecha del trabajo. (Para la definición de *más parecido*, véase después.) Utilice el precio de compra subastado del trabajo más parecido como precio base. El precio de compra máximo se encuentra agregando 8% al precio base, compuesto anualmente, para cada año desde la subasta.
- **Trabajo maestro.** Primero, calcule el precio de compra máximo como si se tratase de una obra maestra del mismo artista. Después, si el cuadro se pintó

en el siglo  $xxi$ , multiplique esto por 0.25, de lo contrario multiplíquelo por  $(21 - c)/(22 - c)$ , donde  $c$  es el siglo en el cual se pintó el trabajo ( $12 < c < 21$ ).

- **Otra pintura.** Medir las dimensiones del cuadro. El precio de compra máximo está dado por la fórmula  $F \times A$ , donde  $F$  es una constante para ese artista (**coeficiente de moda**) y  $A$  es el área del cuadro en centímetros cuadrados. Si no existe coeficiente de moda para ese artista, Osbert no comprará la pintura.

En el caso de obras maestras y trabajos maestros, el coeficiente de **parecido** entre dos pinturas se calcula como sigue:

Otorgue 1 punto si se trata del mismo medio; de lo contrario, 0.

Otorgue 1 punto si se trata del mismo tema; de lo contrario, 0.

Sume estos dos números, multiplíquelos por el área de la más pequeña de las dos pinturas y divida por el área de la más grande.

El número resultante es el coeficiente de parecido.

Si el coeficiente de parecido entre la pintura bajo consideración y todas las pinturas dentro del archivo de información de subastas es 0, entonces Osbert no comprará el trabajo maestro u obra maestra.

El producto de software debe incluir una lista de artistas y sus valores  $F$  correspondientes. El valor de  $F$  puede variar de un mes a otro, dependiendo de qué tan de moda esté un artista actualmente, así que la lista debe implementarse de tal forma que permita a Osbert realizar actualizaciones regulares. Osbert por sí mismo determina el valor de  $F$  con base en su conocimiento y experiencia y modifica el coeficiente si observa un incremento o reducción en los precios pagados por el trabajo de un artista en particular en informes de ventas de arte en periódicos y revistas o en Internet.

El producto de software debe utilizar información de las subastas de obras maestras durante los últimos 25 años alrededor del mundo. Esta información incluye el nombre del artista, título de la pintura, fecha de la pintura, fecha de subasta, precio de venta y tipo de trabajo. El tipo de trabajo consiste en tres componentes: medio (óleo, acuarela u otro medio), dimensiones (alto y ancho) y tema (retrato, naturaleza muerta, paisaje u otro tema). Cada mes Osbert recibe un CD con actualizaciones de precios de subasta alrededor del mundo; estos precios nunca los modifica Osbert.

El equipo de requerimientos actualiza las descripciones de los casos de uso para reflejar esta información. La descripción resultante del caso de uso Comprar una pintura se muestra en la figura 10.14 y del caso de uso Vender una pintura en la figura 10.15.

Ahora, se concentran en los informes. Existen tres informes: compras durante el año pasado, ventas durante el año pasado y detección de nuevas tendencias. De nuevo, observan los registros manuales de Osbert, incluyendo los informes que actualmente genera. Determinan que sus necesidades son las siguientes (en todos los informes se deben incluir signos de interrogación en el nombre o apellido del artista o en el título o fecha del trabajo):

Se necesita un informe para desplegar todas las pinturas compradas durante el año pasado. La salida debe estar en el siguiente orden:

Clasificación (obra maestra, trabajo maestro, otra pintura)  
Fecha de compra  
Apellido del artista  
Título de la pintura  
Precio de compra máximo sugerido  
Precio de compra real.

Cualquier pintura comprada a un precio de compra mayor que el precio de compra máximo calculado por el algoritmo deberá marcarse (colocando un asterisco antes de la clasificación). Este informe debe ordenarse por clasificación y por fecha de compra dentro

### Descripción breve

El caso de uso Comprar una pintura permite a Osbert Oglesby comprar una pintura.

### Descripción paso a paso

1. Osbert introduce los detalles de la pintura que está considerando comprar. Éstos son:
  - Nombre del artista
  - Apellido del artista
  - Título del trabajo
  - Año del trabajo
  - Clasificación (obra maestra, trabajo maestro, otro)
  - Altura
  - Ancho
  - Tema ( retrato, naturaleza muerta, paisaje, otro)
2. El producto de software responde con el máximo precio de compra que Osbert debe ofrecer.
  - 2.1 Para una obra maestra, el producto de software calcula el coeficiente de parecido entre cada pintura del artista por el cual existe un registro de subasta y la pintura en consideración para compra. Los signos de interrogación en el nombre o apellido del artista o en el título o en la fecha del trabajo se ignoran.
    - El producto de software agrega 1 punto cuando se trata del mismo medio, de lo contrario 0.
    - El producto de software agrega 1 punto cuando se trata del mismo tema, de lo contrario 0.
    - El producto software suma estos puntos, multiplica por el área de la pintura más pequeña y divide entre el área de la pintura más grande.
    - El número resultante es el coeficiente de parecido.
  - 2.2 El producto de software encuentra la pintura subastada con el coeficiente de similaridad más grande diferente de cero. Si no existe tal pintura, Osbert no comprará la pintura considerada.
  - 2.3 El producto de software calcula el máximo precio de compra agregando al precio de la subasta del trabajo más parecido 8.5%, compuesto anualmente, para cada año desde la subasta.
3. Si Osbert compra la pintura, introduce más detalles. Éstos son:
  - Fecha de compra
  - Nombre del vendedor
  - Dirección del vendedor
  - Precio de compra real
4. Después el producto de software registra lo siguiente:
  - Precio de compra máximo determinado por el algoritmo
  - Precio de venta deseado

**FIGURA 10.14**

La descripción del caso de uso Comprar una pintura de la revisión de requerimientos del caso de estudio de Osbert Oglesby.

de esa clasificación. La tasa promedio del precio de compra real contra el precio de compra máximo sugerido por el algoritmo para todas las pinturas debe desplegarse en el informe al final de éste. Un ejemplo de dicho informe se muestra en la figura 10.16.

#### Descripción breve

El caso de uso Vender una pintura permite a Osbert Oglesby vender una pintura.

#### Descripción paso a paso

- Osbert ingresa los detalles de la pintura que vendió. Éstos son:

Fecha de venta  
Nombre del comprador  
Dirección del comprador  
Precio de venta real

**FIGURA 10.15** La descripción del caso de uso Vender una pintura de los requerimientos revisados del caso de estudio Osbert Oglesby.

Fecha del informe: 08/14/2004		
Osbert Oglesby - coleccionista de arte fino		
O		
F : Obra aestra	F : O : 02/08/2004	
O: at ayar	O: able o ntain	
O : 4 00	O : 0 : 12 000	
<hr/>		
F : Obra aestra	F : O : 0 / 0 / 2004	
O: at ayar	O: al er ay	
O : 1 80	O : 0 : 000	
<hr/>		
F : Otra	F : O : 04/10/2004	
O: mman	O: les and Oranges	
O : 1 400	O : 0 : 2 000	
<hr/>		
asa romedio: 1		

**FIGURA 10.16** Un ejemplo del informe que muestra las pinturas compradas por Osbert Oglesby durante el año pasado.

Un segundo informe debe desplegar las pinturas que se han vendido durante el último año. La salida debe mostrarse en el siguiente orden:

Clasificación  
Fecha de venta  
Apellido del artista  
Título de la pintura  
Precio de venta deseado  
Precio de venta real.

Cualquier pintura vendida a un precio de 5% o más por debajo de precio de venta deseado deberá marcarse (colocando un asterisco antes de la clasificación). Este informe debe ser ordenado por la clasificación y por la fecha de venta dentro de la clasificación. La tasa

Fecha del informe: 08/14/2004 Osbert Oglesby - coleccionista de arte fino			
F : Obra aestra	O: at ayar	F : 0 /11/2004	O: able o ntain
O : 2 4 4 0		O : 2 0 000	
F : Obra aestra	O: at ayar	F : 0 /12/2004	O: al er ay
O : 1 1 2 0		O : 20 000	
F : Otra	O: mann	F : 0 /1 /2004	O: les and Oranges
O : 4 00		O : 00	
asa romedio: 1 11			

**FIGURA 10.17** Un ejemplo del informe que muestra las pinturas que vendió Osbert Oglesby durante el año pasado.

promedio del precio de venta real contra el precio de venta deseado para todas las pinturas en el informe deberá desplegarse al final del informe. Nuevamente, se muestra un informe de ejemplo en la figura 10.17.

Por último, Osbert está ansioso de detectar las tendencias nuevas en el mercado del arte tan pronto como sea posible. Él está particularmente interesado en determinar cuándo se pagarán mayores precios que los esperados por el trabajo de un artista en particular, de esta forma él puede comprar las pinturas de dicho artista antes de que otros se percaten de la tendencia. Por esto mismo él requiere un informe que muestre a los artistas cuyos trabajos ha vendido a un precio que ha excedido el precio de venta deseado en cada instancia durante el año pasado. Para que un artista aparezca en este informe, Osbert debió haber vendido al menos dos de sus trabajos durante el periodo. Los nombres de los artistas relevantes (si es que existen) deberán estar en orden alfabético. Cada nombre debe aparecer en una línea

Fecha del informe: 08/14/2004 Osbert Oglesby - coleccionista de arte fino			
TENDENCIAS			
Artista: Da id at ayar			
C ASIFICACI N: Obra aestra	FEC A DE ENTA: 0 /11/2004	T T O: Table o ntain	ECIO DE ENTA: 2 0 000
FECCIO DESEADO: 2 4 4 0			
C ASIFICACI N: Obra aestra	FEC A DE ENTA: 0 /12/2004	T T O: al er ay	ECIO DE ENTA: 20 000
FECCIO DESEADO: 1 1 2 0			

**FIGURA 10.18** Un ejemplo del informe que muestra las nuevas tendencias en el mercado del arte.

**Descripción breve**

El caso de uso Generar un informe le permite a Osbert Oglesby obtener información de las pinturas que compró o vendió el año pasado o de detectar nuevas tendencias en el mercado del arte.

**Descripción paso a paso**

1. Los siguientes informes deben ser generados bajo demanda (los signos de interrogación en el nombre o en el apellido del artista, o en el título o en la fecha de trabajo deben imprimirse):

**1.1 Informe de compras**

El sistema despliega todas las pinturas compradas durante el año pasado.

La salida es en este orden:

Clasificación  
Fecha de compra  
Apellido del artista  
Título de la pintura  
Precio de compra máximo sugerido  
Precio de compra real

Cualquier pintura comprada por más del precio de compra máximo calculado por el algoritmo se marca (colocando un asterisco antes de la clasificación).

Este informe se ordena por clasificación y por la fecha de compra dentro de la clasificación.

La tasa promedio del precio de compra real contra el precio de compra máximo sugerido por el algoritmo para todas las pinturas dentro del informe se despliega al final de éste.

**1.2 Informe de ventas**

El sistema despliega todas las pinturas vendidas durante el año pasado.

La salida es en este orden:

Clasificación  
Fecha de venta  
Apellido del artista  
Título de la pintura  
Precio de venta deseado  
Precio de venta real

Cualquier pintura vendida a un precio de 5% o más por debajo del precio de venta deseado se marca (colocando un asterisco antes de la clasificación).

Este informe se ordena por clasificación y por la fecha de venta dentro de la clasificación.

La tasa promedio del precio de venta real contra el precio de venta deseado para todas las pinturas dentro del informe se despliega al final de éste.

**1.3 Informe de tendencias futuras**

El sistema despliega una lista de todos los artistas cuyos trabajos se hayan vendido a un precio que excedió el precio de venta deseado en cualquier momento durante el año pasado.

Para que un artista aparezca en este informe, al menos dos de sus trabajos se debieron haber vendido en dicho periodo.

Los nombres de los artistas relevantes (si es que existen) están en orden alfabético. Cada nombre aparece en una línea de la pantalla. Los distintos trabajos vendidos aparecen en líneas sucesivas, en orden de la fecha de venta.

Para cada trabajo, la salida es en este orden:

Clasificación  
Título de la pintura  
Fecha de venta  
Precio de venta deseado  
Precio de venta real

**FIGURA 10.19** La descripción del caso de uso Generar un informe de los requerimientos revisados del caso de estudio Osbert Oglesby.

de la pantalla. Los distintos trabajos de dicho artista que se hayan vendido deberán entonces aparecer en el orden de la fecha de venta. Para cada trabajo, debe aparecer lo siguiente:

- Clasificación
- Título de la pintura
- Fecha de venta
- Precio de venta deseado
- Precio de venta real

Un ejemplo del informe se muestra en la figura 10.18.

La descripción actualizada del caso de uso Generar un informe, que incorpora estos detalles, aparece en la figura 10.19.

## Caso de estudio

### 10.10 El flujo de trabajo de las pruebas: el caso de estudio Osbert Oglesby

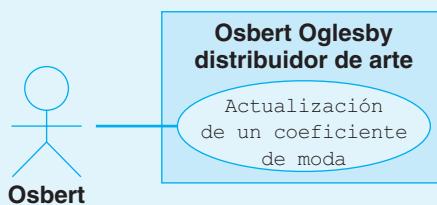
Mientras se realiza el flujo de trabajo de los requerimientos, como con cualquier otro desarrollo de software y actividad de mantenimiento, todos los miembros del equipo de requerimientos tienen que revisar continuamente su trabajo. Ahora que el flujo de trabajo de los requerimientos parece estar completo, los artefactos del flujo de trabajo de requerimientos necesitan revisarse por miembros del grupo SQA. En otras palabras, el **flujo de trabajo de las pruebas** ahora se lleva a cabo.

El grupo SQA descubre una omisión seria: un caso de uso no se ha considerado. El siguiente párrafo aparece en la sección 10.9:

El producto de software debe incluir una lista de artistas y sus valores  $F$  correspondientes. El valor de  $F$  puede variar de un mes a otro, dependiendo de qué tan de moda esté un artista actualmente, así que la lista debe implementarse de tal forma que permita a Osbert realizar actualizaciones regulares.

En otras palabras, los miembros del equipo de requerimientos olvidaron incluir un caso de uso para actualizar un **coeficiente de moda**. Por esto mismo ellos agregan el caso de uso Actualización de un coeficiente de moda de la figura 10.20 y su descripción (figura 10.21). La segunda iteración del diagrama de casos de uso, que incorpora el caso de uso faltante, se muestra en la figura 10.22.

Después de seguir revisando, en apariencia todo es satisfactorio, al menos por ahora. Sin embargo, durante el desarrollo del análisis orientado a objetos vendrán a la luz nuevas fallas, se podrán necesitar requerimientos adicionales o los existentes pudieran no utilizarse.



**FIGURA 10.20**

El caso de uso Actualización de un coeficiente de moda de los requerimientos revisados del caso de estudio Osbert Oglesby.

El paradigma orientado a objetos es iterativo e incremental y los miembros del equipo de desarrollo siempre verificarán qué cambios y extensiones a la versión actual del producto de software tendrán que hacerse en cualquier momento.

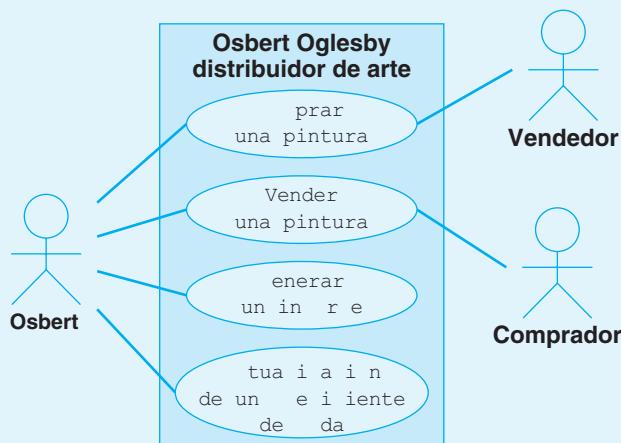
#### Descripción breve

El caso de uso Actualización de un coeficiente de moda le permite a Osbert Oglesby modificar el coeficiente de moda F para un artista.

#### Descripción paso a paso

1. Osbert ingresa el nuevo coeficiente de moda para el artista.

**FIGURA 10.21** La descripción del caso de uso Actualización de un coeficiente de moda de los requerimientos revisados del caso de estudio Osbert Oglesby.



**FIGURA 10.22** La segunda iteración del diagrama de casos de uso del caso de estudio Osbert Oglesby.

Esto concluye la descripción del flujo de trabajo de los requerimientos para el caso de estudio Osbert Oglesby.

## 10.11 La fase de requerimientos clásica

Por un lado, no existe tal cosa como “requerimientos orientados a objetos”, tampoco debe haber tal cosa. El propósito del flujo de trabajo de los requerimientos es determinar las necesidades del cliente, esto es, qué funcionalidad debe tener el sistema deseado. El flujo de trabajo de los requerimientos no tiene nada que ver con cómo se construye el producto. Desde este punto de vista, no tiene sentido referirse al paradigma clásico o al paradigma orientado a objetos dentro del contexto del flujo de trabajo de los requerimientos, no más de lo que se puede referir al manual de usuario clásico u orientado a objetos. Después de todo, el manual de usuario describe los pasos a seguir por el usuario cuando ejecute el producto de software y no tiene nada que ver con cómo se construyó el producto.

De la misma forma, el flujo de trabajo de los requerimientos genera una propuesta sobre lo que realiza el producto; la forma en que el producto se construirá no cabe aquí.

Por otro lado, el enfoque completo de las secciones 10.2 a 10.10 es orientado a objetos en la naturaleza en que se orienta al modelo. Los casos de uso, junto con sus descripciones, forman la base del flujo de trabajo de los requerimientos. Tal como se muestra a lo largo de la parte 2 de este libro, modelar es el objetivo del paradigma orientado a objetos.

Sin embargo, modelar en general (y el modelado de UML en particular) no es una parte del paradigma clásico. La fase de los requerimientos clásica comienza con la generación de requerimientos seguida por el análisis de requerimientos, similar al paradigma orientado a objetos (secciones 10.3 a 10.4.2). Pero desde ese punto en adelante, los dos paradigmas difieren. En lugar de construir modelos, el próximo paso en la fase de requerimientos clásica es trazar una lista de requerimientos. El paso usual después es construir un prototipo rápido que implemente las funcionalidades clave sustentando esos requerimientos, tal como se describe en la sección 2.9.3. El cliente y los usuarios futuros del producto de software deseado enseguida experimentarán con el prototipo rápido hasta que los miembros del equipo de requerimientos estén satisfechos de que el prototipo rápido exhiba las funcionalidades clave del producto de software que el cliente necesita.

Construir un prototipo rápido para el producto como un todo no forma parte del paradigma orientado a objetos, por las razones dadas en la sección 12.17. Sin embargo, es recomendable construir un prototipo rápido de la interfaz de usuario, tal como se describirá.

## 10.12 Elaboración rápida de un prototipo

Un **prototipo rápido** es un software desarrollado precipitadamente que exhibe la funcionalidad clave del producto deseado. Por ejemplo, un producto que ayuda a administrar un edificio de apartamentos debe incorporar una pantalla de entrada que permita al usuario ingresar los detalles del nuevo inquilino e imprimir un informe de ocupación para cada mes. Estos aspectos se incorporan en el prototipo rápido. Sin embargo, las capacidades para revisar errores, rutinas de actualización de archivos y cálculos complejos de impuestos quizás no se incluyan. El punto clave es que el prototipo rápido refleja la funcionalidad que el cliente ve, tal como pantallas de entrada e informes, pero omite aspectos “escondidos”, como la actualización de archivos. (Para revisar de otra manera los prototipos rápidos véase *Por si desea saber*, cuadro 10.3.)

El cliente y los usuarios a los que se dirige el producto ahora experimentan con el prototipo rápido, mientras que miembros del equipo de desarrollo observan y toman notas. Con base en sus experiencias, los usuarios comentan a los desarrolladores la forma en que el prototipo rápido satisface sus necesidades y, más importante, identifican las áreas que necesitan mejorarse. Los desarrolladores modifican el prototipo rápido hasta que ambos bandos se convencen de que las necesidades del cliente están correctamente encapsuladas en el prototipo rápido. Después se usa el prototipo rápido como base para trazar las especificaciones.

Un aspecto importante del modelo de prototipo rápido se encuentra en la palabra *rápido*. La idea es construir un prototipo rápido tan pronto como sea posible. Después de todo, el propósito de un prototipo rápido es proporcionar al cliente una comprensión del producto, y mientras más pronto mejor. No importa si el prototipo rápido apenas trabaja, si se colapsa cada cierto tiempo, o si los despliegues de pantalla son menos que perfectos. El propósito del prototipo rápido es permitir al cliente y a los desarrolladores ponerse de acuerdo tan rápido como se pueda en lo que el producto debe efectuar. Por esto, cualquier imperfección en el prototipo rápido se puede ignorar, teniendo en cuenta no afectar seriamente la funcionalidad del prototipo rápido y que por esto den una impresión incorrecta de cómo se comporta el producto.

Un segundo aspecto importante del modelo de prototipo rápido es que éste debe construirse para modificarse. Si la primera versión del prototipo rápido no es lo que el cliente necesita, entonces el prototipo debe transformarse rápidamente en una segunda versión que, se espera, satisfaga mejor los requerimientos del cliente. Para lograr un desarrollo rápido a lo largo del proceso del prototipo rápido, lenguajes de cuarta generación (4GL) y lenguajes interpretados como Smalltalk, Prolog y Lisp se han utilizado con el propósito de realizar un prototipo rápido. Lenguajes populares de prototipo rápido de hoy en día incluyen HTML y Perl, así como Visual C++ y Visual J++. Se han

### CUADRO 10.3

Desde hace mucho tiempo prevalece la idea de construir modelos que muestren los aspectos clave de un producto. Por ejemplo, una pintura de 1618 de Domenico Cresti (conocido como "Il Passignano" debido a que él nació en el pueblo de Passignano en la región Chianti de Italia) muestra a Miguel Ángel presentando al papa Pablo IV un modelo de madera que diseñó para San Pedro (en Roma). Tales modelos de arquitectura pudieran ser enormes; un modelo de una propuesta de diseño anterior para San Pedro hecha por el arquitecto Bramante tiene más de 20 pies de largo de cada lado.

Los modelos de arquitectura se usaban para distintos propósitos. Primero, como se ilustra en la pintura Cresti (ahora colgando en la Casa Buonarroti en Florencia), los modelos se usaban para intentar que el cliente se interesara en fondear el proyecto. Esto es semejante al uso de un prototípico rápido que determine las necesidades reales del cliente. Segundo, en una era anterior a los dibujos arquitectónicos, el modelo mostraba al constructor la estructura del edificio y a los albañiles les indicaba cómo debería ser decorado el edificio. Esto es similar a la forma en que ahora construimos un prototípico rápido de la interfaz de usuario, tal como se describe en la sección 10.13.

No es una buena idea, sin embargo, dibujar un paralelismo tan cercano entre dichos modelos arquitectónicos y los prototípicos rápidos de software. Estos últimos se utilizan durante la fase de requerimientos clásica para extraer las necesidades del cliente. A diferencia de los modelos arquitectónicos, no se utilizan para representar ni el diseño de arquitectura ni el diseño detallado; el diseño se genera en dos fases posteriores, esto es, durante la fase de diseño clásica.

Por si desea saber

expresado preocupaciones acerca del mantenimiento de ciertos lenguajes interpretados, pero desde el punto de vista del prototípico rápido es irrelevante. Todo lo que cuenta es esto: ¿puede un lenguaje dado ser utilizado para generar un prototípico rápido? y ¿Puede el prototípico rápido modificarse rápidamente? Si la respuesta a ambas preguntas es Sí, entonces dicho lenguaje es un buen candidato para prototípico rápido.

Realizar un prototípico rápido es particularmente efectivo cuando se desarrolla la interfaz de usuario de un producto. Este uso se comenta en la siguiente sección.

## 10.13 Factores humanos

Es importante que tanto el cliente como los usuarios futuros del producto interactúen con el prototípico rápido de la interfaz de usuario. Motivar a los usuarios a experimentar con la interfaz humano-computadora (HCI) reduce en gran medida el riesgo de que el producto final tenga que alterarse. En particular, este experimento ayuda a conseguir una interfaz amigable al usuario, un objetivo vital de todos los productos de software.

El término **amigable al usuario** se refiere a la facilidad con la cual los seres humanos pueden comunicarse con el producto de software. Si los usuarios tienen dificultad en aprender cómo utilizar un producto o encuentran las pantallas confusas o irritantes, entonces o no usarán el producto o lo usarán incorrectamente. Para tratar de eliminar este problema, se introdujeron productos que se manejan con menús. En lugar de tener que digitar un comando tal como **realizar cálculo** o **imprimir el informe de tasa de servicio**, el usuario únicamente tiene que seleccionar de un conjunto de respuestas posibles, como

1. Realizar cálculo
2. Imprimir informe de tasa de servicio
3. Seleccionar vista a graficar

En este ejemplo, el usuario ingresa 1, 2 o 3 para invocar el comando correspondiente.

En estos días, en lugar de simplemente desplegar líneas de texto, las HCI emplean gráficas. Las ventanas, los iconos y los menús desplegables son componentes de una **interfaz gráfica de usuario** (GUI, por sus siglas en inglés). Debido a los muchos sistemas de ventanas, han surgido estándares como X Window. También, la selección **apuntar y hacer clic** es ahora una norma. El usuario mueve un ratón (esto es, un dispositivo de señalización) para deslizar el cursor en la pantalla hacia la respuesta deseada (“apuntar”) y presiona un botón del ratón (“clic”) para seleccionar una respuesta.

Sin embargo, aunque el producto deseado emplee tecnología moderna, los diseñadores nunca deben olvidar que el producto será usado por seres humanos. En otras palabras, los diseñadores del HCI deben considerar **factores humanos**, tales como el tamaño de las letras, mayúsculas-minúsculas, color, largo de línea y número de líneas en la pantalla.

Otro ejemplo de factores humanos aplica al menú precedente. Si el usuario escoge la opción **3. Seleccionar vista a graficar**, entonces aparece otro menú con otra lista de opciones. A menos que se diseñe bien un sistema manejable a través de menús, existe el peligro de que los usuarios se encuentren con una secuencia larga de menús para realizar una operación relativamente simple. Este retraso puede molestar a los usuarios, algunas veces les puede causar que hagan selecciones inapropiadas en el menú. También, el HCI debe permitir al usuario modificar una selección previa sin tener que regresar al menú superior y comenzar de nuevo. Este problema puede existir incluso cuando se use una GUI debido a que muchas interfaces gráficas de usuario son esencialmente una serie de menús desplegados en un formato atractivo en la pantalla.

Algunas veces es imposible que una interfaz de usuario sencilla complazca a todos los usuarios. Por ejemplo, si tanto profesionales en computación como pasantes de preparatoria sin experiencia previa en computación deben usar un producto, entonces es preferible diseñar dos conjuntos distintos de HCI, cada uno dirigido cuidadosamente al nivel de habilidad y perfil psicológico de los usuarios a los que va dirigido. Esta técnica se puede extender incorporando conjuntos de interfaces de usuario que requieran diferentes niveles de sofisticación. Si el producto deduce que el usuario estará más confortable con una interfaz de usuario menos sofisticada, tal vez es porque el usuario esté cometiendo errores frecuentes o esté invocando continuamente la ayuda, entonces se le muestran al usuario automáticamente pantallas que son más apropiadas para su nivel de habilidad actual. Pero, de acuerdo a cómo se familiarice el usuario con el producto se despliegan pantallas minimalistas que proporcionen menos información, para obtener un final más rápido. Este enfoque automatizado reduce la frustración del usuario e incrementa la productividad [Schach y Wood, 1986].

Muchos beneficios pueden acrecentarse cuando se toman en cuenta los factores humanos durante el diseño de una HCI, incluyendo tiempos reducidos de aprendizaje y menores tasas de error. Aunque siempre se debe proporcionar ayuda, ésta es menos con un diseño cuidadoso de HCI. Esto también incrementa la productividad. La uniformidad en la apariencia de la HCI a lo largo de un producto o grupo de productos puede ocasionar conocimiento intuitivo en los usuarios sobre cómo usar una pantalla que nunca habían visto antes puesto que es similar a otras pantallas con las que están familiarizados. Los diseñadores del software Macintosh han tomado este principio en cuenta; ésta es una de las muchas razones por las cuales el software para Macintosh generalmente es tan amigable.

Se sugiere que todo lo que se necesita para diseñar una HCI amigable es el simple sentido común. Ya sea que esto sea o no cierto, es esencial que sea construido un prototipo rápido del HCI de cada producto. Los usuarios a los que va dirigido el producto pueden experimentar con el prototipo rápido del HCI e informar a los diseñadores si el producto deseado en verdad es amigable, esto es, si los diseñadores tomaron en cuenta los factores humanos necesarios.

En la siguiente sección se comenta la reutilización dentro del contexto del prototipo rápido.

## 10.14 Reutilización del prototipo rápido

Después de que se construyó el prototipo rápido, se elimina muy temprano del proceso de software. Una forma alterna pero generalmente tonta de proceder es desarrollar y detallar el prototipo rápido hasta que éste se convierta en el producto. En teoría, este enfoque conducirá a un desarrollo de software rápido; después de todo, en lugar de tirar el código que forma parte del prototipo rápido, junto con el conocimiento construido en el mismo, el prototipo rápido se convierte en el producto final. Sin

embargo, en la práctica, el proceso es muy similar al enfoque codificar y reparar que se muestra en la figura 2.7. Así, como con el modelo codificar y reparar, el primer problema con esta forma de modelo de prototipo rápido surge del hecho de que, en el curso de detallarlo, se deben realizar cambios sobre un producto operacional. Ésta es una forma cara de proceder, como se muestra en la figura 1.6. Un segundo problema es que uno de los principales objetivos cuando se construye un prototipo rápido es la velocidad al construir. Un prototipo rápido es (y así debe serlo) apresuradamente elaborado, en lugar de especificado, diseñado e implementado con cuidado. Sin una especificación y documentos de diseño, el código resultante es difícil y caro de mantener. Parecería ser un desperdicio construir un prototipo rápido para después tirarlo y diseñar el producto desde cero, pero es mucho más barato tanto a corto como a largo plazos hacer esto en lugar de intentar convertir el prototipo rápido en un software de calidad [Brooks, 1975].

Otra razón para descartar un prototipo rápido es el elemento desempeño, en particular en sistemas en tiempo real. Para asegurar que las restricciones de tiempo se cumplan, es necesario diseñar cuidadosamente el producto. En contraste, un prototipo rápido se construye para desplegar la funcionalidad clave al cliente; no se controlan los elementos de desempeño. Como resultado, si se intenta detallar un prototipo rápido para convertirlo en un producto para entregar, quizás no se cumplan los tiempos de respuesta y otras restricciones de tiempo.

Una forma de asegurar que el prototipo rápido se tire y se diseñe e implemente con propiedad el producto es construir el prototipo rápido en un lenguaje distinto al del producto. Por ejemplo, el cliente pudiera especificar que el producto se escriba en Java. Si el prototipo rápido se implementa en HTML, por ejemplo, debe ser eliminado. Primero, el prototipo rápido se implementa en HTML y se detalla hasta que el cliente está satisfecho de que cumple con todo, o casi todo, lo que el producto deseado realizará. Segundo, el producto se diseña con base en el conocimiento y las habilidades adquiridas al construir el prototipo rápido. Por último, el diseño se implementa en Java y el producto probado se entrega al cliente de la manera acostumbrada.

Sin embargo, existe un caso en el que se permite detallar un prototipo rápido o, más específicamente, porciones del prototipo rápido. Cuando se generan porciones del prototipo rápido por computadora, dichas porciones pueden usarse en el producto final. Por ejemplo, las interfaces de usuario por lo común son un aspecto clave de un prototipo rápido. Cuando las herramientas CASE como generadores de pantallas y generadores de informes (sección 5.5) se utilizan para generar las interfaces de usuario, esas porciones del prototipo rápido pueden en verdad utilizarse como una parte del software de calidad.

El deseo de no “desperdiciar” el prototipo rápido ha resultado en una versión modificada del modelo de prototipos rápidos que algunas organizaciones han adoptado. Aquí, la administración, *antes* de que se construya el prototipo rápido, decide qué porciones se pueden utilizar en el producto final, tomando en cuenta que esas porciones pasen las mismas pruebas para asegurar la calidad como lo hacen otros componentes de software. Por esto mismo, después de que el prototipo rápido está completo, dichas secciones que los desarrolladores desean seguir usando deben pasar las inspecciones de diseño y codificación. Este enfoque va más allá del prototipo rápido. Por ejemplo, los componentes que tienen la suficiente calidad para pasar las inspecciones de diseño y codificación generalmente no se encuentran en un prototipo rápido. Además, los documentos de diseño no forman parte de un prototipo rápido clásico. Sin embargo, este enfoque híbrido les atrae a algunas organizaciones que esperan recuperar parte del tiempo y dinero invertido en el prototipo rápido. Sin embargo, para asegurar que la calidad del código sea lo suficientemente alta, el prototipo rápido debe construirse un poco más lento que un prototipo “rápido” ordinario.

## 10.15 Herramientas CASE para el flujo de trabajo de los requerimientos

Todos los diagramas UML de este capítulo reflejan la importancia de tener una herramienta gráfica que ayude con el flujo de trabajo de los requerimientos. Esto es, lo que se necesita es una herramienta de dibujo que le permita al usuario dibujar diagramas UML relevantes con facilidad. Tal herramienta posee dos grandes fortalezas. Primero, mientras se itera generalmente es más fácil modificar un diagrama almacenado en dicha herramienta que volver a dibujar el diagrama a mano. Segundo, cuando

se usa una herramienta CASE de este tipo, los detalles del producto se almacenan en la herramienta CASE. Por lo mismo la documentación siempre está disponible y al día.

Una debilidad de dichas herramientas CASE es que no siempre son amigables. Un ambiente gráfico poderoso tiene tanta funcionalidad que generalmente tiene una curva de aprendizaje inclinada, e incluso usuarios experimentados algunas veces tienen dificultad para recordar cómo realizar una tarea en particular. Una segunda debilidad es que es casi imposible programar una computadora para que dibuje diagramas UML que sean tan bien diseñados como los diagramas dibujados a mano por humanos. Una alternativa es invertir una cantidad considerable de tiempo “ajustando” un diagrama creado con una herramienta. Sin embargo, este enfoque es algunas veces tan lento como dibujar los diagramas a mano. Peor aún, las restricciones de muchas herramientas CASE gráficas son tales que no importa cuánto tiempo y esfuerzo se haya puesto en el diagrama, nunca se verá tan bien como un diagrama dibujado a mano. Un tercer problema es que muchas herramientas CASE son caras. No es inusual tener que pagar \$5 000 o más por usuario por una herramienta CASE. Por otro lado, existe una cantidad de herramientas CASE de código abierto de este tipo que se pueden descargar sin costo. En resumen, las dos fortalezas de las herramientas CASE mostradas en el primer párrafo de esta sección sobreponen estas debilidades.

Muchos de los ambientes gráficos de herramientas CASE clásicos, como System Architect y Software through Pictures se han extendido para dar soporte a diagramas UML. Además, existen ambientes de herramientas CASE orientados a objetos, como Rose y Together. También existen herramientas CASE de código abierto de este tipo, incluyendo ArgoUML.

## 10.16 Métricas para el flujo de trabajo de los requerimientos

Una función clave del flujo de trabajo de los requerimientos es qué tan rápido el equipo de requerimientos determina las necesidades reales del cliente. Por esto una métrica útil durante el flujo de trabajo constituye una medida de la volatilidad de los requerimientos. Mantener un registro de qué tan frecuentemente se modifican los requerimientos durante el flujo de trabajo de los requerimientos otorga a la administración una forma de determinar la tasa a la cual el equipo de requerimientos converge con los requerimientos reales del producto. Esta métrica tiene la ventaja adicional de que puede aplicarse a cualquier técnica de extracción de requerimientos, como entrevistas o análisis de formas.

Otra medida de qué tan bien el equipo de requerimientos está haciendo su trabajo es el número de requerimientos que se modifican durante el resto del proceso de desarrollo de software. Por cada modificación en los requerimientos, se debe registrar ya sea que el cambio lo inició el cliente o los desarrolladores. Si los desarrolladores inician un gran número de requerimientos durante el análisis, diseño y flujos de trabajo subsecuentes, entonces queda claro que el proceso que el equipo utiliza para llevar a cabo el flujo de trabajo de requerimientos debe ser revisado con cuidado. De manera inversa, si el cliente hace modificaciones en varias ocasiones sobre los requerimientos durante los flujos de trabajo subsecuentes, entonces esta métrica puede usarse para advertir al cliente que el problema del blanco móvil puede afectar seriamente el proyecto y que los futuros cambios deberían mantenerse al mínimo.

## 10.17 Retos del flujo de trabajo de los requerimientos

Como cualquier otro flujo de trabajo del proceso de desarrollo de software, existen problemas potenciales y obstáculos asociados con el flujo de trabajo de los requerimientos. Primero, es esencial tener la cooperación de los usuarios potenciales del producto deseado desde el principio del proceso. Los individuos por lo común temen que las computadoras les quiten sus trabajos. Existen fundamentos para tener ese miedo. Durante los últimos 30 años, más o menos, el impacto de la sistematización ha reducido la necesidad de empleados sin habilidades, pero también ha generado empleos para trabajadores con habilidades. En resumen, el número de oportunidades de empleo bien pagadas creadas como consecuencia directa de la sistematización ha excedido en mucho el número relativo de empleos sin

habilidades, como lo evidencia tanto la disminución en las tasas de desempleo y el incremento en la compensación promedio. Pero el crecimiento económico sin paralelo de muchos países alrededor del mundo como una consecuencia directa o indirecta de la llamada Era de la Computación de ninguna forma puede compensar el impacto negativo sobre aquellos individuos que perdieron sus empleos como resultado de la sistematización.

Es esencial que cualquier miembro del equipo de requerimientos esté atento en todo momento en que los miembros de la organización cliente con los que ellos interactúen estén profundamente preocupados por el impacto potencial que el producto de software deseado tiene sobre sus empleos. En el peor caso, los empleados pudieran deliberadamente proporcionar información incorrecta o no cooperar, para tratar de asegurar que el producto no cumpla con las necesidades del cliente y, de esta forma, proteger sus empleos. Pero, incluso sin sabotajes de este tipo, algunos miembros de la organización cliente no pudieran ser útiles, simplemente porque se sienten, en forma vaga, amenazados por la sistematización.

Otro reto del flujo de trabajo de requerimientos es la habilidad de **negociar**. Por ejemplo, generalmente es necesario escalar hacia abajo lo que el cliente quiere. Y no resulta sorpresivo que casi a cualquier cliente le encantaría tener un producto de software que pudiera hacer todo lo que pueda imaginar que necesita. Tal producto implicaría un inaceptable y largo tiempo de construcción y costaría mucho más de lo que el cliente considera razonable. Por esto, casi siempre es necesario persuadir al cliente para que acepte menos (muchas veces mucho menos) de lo que él o ella quieren. Calcular los costos y beneficios (véase sección 5.2) de cada requerimiento en disputa puede ayudar en este aspecto.

Otro ejemplo de la necesidad de negociar es la habilidad de llegar a un compromiso entre gerentes respecto a la funcionalidad del producto deseado. Por ejemplo, un gerente astuto pudiera intentar extender su poder incluyendo un requerimiento que puede implementarse únicamente incorporando a su área de responsabilidad ciertas funciones de negocio que actualmente son responsabilidad de otro gerente. Es obvio que, al descubrir lo anterior, el otro gerente objetará fuertemente. El equipo de requerimientos debe sentarse con ambos gerentes y resolver el problema.

Un tercer reto del flujo de trabajo de los requerimientos es que en muchas organizaciones los individuos que poseen la información que el equipo de requerimientos necesita extraer, simplemente no tienen tiempo para reunirse para discusiones en detalle. Cuando esto sucede, el equipo debe informar al cliente, quien entonces deberá decidir qué es más importante, las responsabilidades actuales del trabajo de los individuos o la construcción del producto de software. Y, si el cliente decide que el producto de software no es lo más importante, los desarrolladores pueden no tener más alternativa que retirarse de un proyecto destinado al fracaso.

Por último, la flexibilidad y objetividad son esenciales para extraer los requerimientos. Es vital que los miembros del equipo de requerimientos enfoquen cada entrevista sin prejuicios. En particular, un entrevistador nunca debe asumir algo acerca de los requerimientos como resultado de entrevistas previas, y luego conducir entrevistas subsecuentes dentro del marco de esos supuestos. Por el contrario, un entrevistador debe en forma consciente suprimir cualquier información recavada en entrevistas previas y conducir cada entrevista de manera imparcial. Hacer supuestos prematuros referentes a los requerimientos es peligroso; asimismo, hacer cualquier supuesto durante el flujo de trabajo de requerimientos acerca del software a construir puede ser desastroso.

El capítulo concluye con el cuadro 10.1, el cual resume los pasos del flujo de trabajo de los requerimientos.

#### CUADRO 10.1 Cómo realizar el flujo de trabajo de requerimientos.

- Iteración
  - Obtener una comprensión del dominio.
  - Trazar el modelo de negocio.
  - Trazar los requerimientos.
- Hasta que los requerimientos sean satisfactorios.

## Revisión del capítulo



El capítulo comienza con una descripción de la importancia de determinar las necesidades del cliente (sección 10.1), seguida de un repaso del flujo de trabajo de los requerimientos (sección 10.2). En la sección 10.3 se describe la necesidad de comprender el dominio. En la sección 10.4 se describe cómo trazar el modelo de negocio. En las secciones 10.4.1 y 10.4.2 se discuten las entrevistas y otras técnicas de extracción de requerimientos. El modelo de negocio se modela usando casos de uso, los cuales se introducen en la sección 10.4.3. En la sección 10.5 se describe el trazo de los requerimientos iniciales. En la sección 10.6 se describe cómo obtener una comprensión inicial del dominio, mientras que en las secciones 10.7 y 10.8 se muestran, respectivamente, el modelo de negocio inicial y los requerimientos iniciales. En la sección 10.9 se detallan los re-

querimientos. En la sección 10.10 se describe el flujo de trabajo de las pruebas para el caso de estudio de Osbert Oglesby. En la sección 10.11 se contrasta la fase de requerimientos clásica con el flujo de trabajo de los requerimientos del Proceso Unificado. Después, en las secciones 10.12 y 10.13, se discute a detalle el prototipo rápido; en la última sección se acentúa la importancia de construir un prototipo rápido para la interfaz de usuario. En la sección 10.14 se hace una advertencia para no reutilizar un prototipo rápido. Después se comentan las herramientas CASE para el flujo de trabajo de requerimientos (sección 10.15) y las métricas para el flujo de trabajo de los requerimientos (sección 10.16). El capítulo concluye con una descripción de los retos de la fase de requerimientos (sección 10.17).

## Lectura adicional



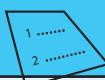
Jackson [1995] es una excelente introducción al análisis de requerimientos. [Thayer y Dorfman, 1999] es una colección de papeles sobre análisis de requerimientos. Berry [2002] sugiere que el efecto de ondas de las modificaciones inevitables a los requerimientos es la razón por la cual no puede haber una ingeniería de software infalible (Por si desea saber, cuadro 3.4). Aparecen artículos sobre requerimientos en enero/febrero 2003 de *IEEE Software*. El uso del análisis costo-beneficio para establecer prioridades entre los requerimientos se describe en Karlsson y Ryan [1997].

El flujo de trabajo de requerimientos del Proceso Unificado se describe con detalle en los capítulos 6 y 7 de [Jacobson, Booch y Rumbaugh, 1999]. Los casos de mal uso (casos de uso que modelan interacciones que el software debe prevenir) de casos se describe en Alexander [2003].

Para una introducción a los prototipos rápidos, los libros sugeridos incluyen [Connell y Shafer, 1989, y Gane, 1989]. El modelo de prototipo rápido es una versión del desarrollo rápido de aplicaciones (RAD); una variedad de artículos sobre RAD están en el volumen de septiembre de 1995 de *IEEE Software*.

Los factores humanos se comentan en Dix, Finlay, Abowd y Beale [1993]; Browne [1994]; y Preece [1994]. Un trabajo clásico sobre el diseño de interfaz de usuario es [Shneiderman, 2003]. Artículos sobre interfaces de usuario pueden encontrarse en los volúmenes de septiembre de 2000 y marzo 2003 de *Communications of the ACM* y en el volumen de marzo de 2002 de *IEEE Computer*. Los acontecimientos de la Conferencia Anual sobre Factores Humanos en Sistemas Computacionales (patrocinados por ACM SIGCHI) son una fuente valiosa de información de un rango amplio de factores humanos.

## Términos clave



- actores, 276
- amigable al usuario, 294
- análisis de requerimientos, 273
- apuntar y hacer clic, 294
- cámaras de video, 275
- captura de requerimientos, 273
- caso de uso, 276
- confiabilidad, 278
- cuestionario, 275
- descripción del caso de uso, 282
- diagrama del caso de uso, 280
- dominio, 272

- dominio de la aplicación, 272
- entrevista estructurada, 274
- entrevista no estructurada, 274
- extracción de requerimientos, 273
- factores humanos, 294
- formas, 275
- flujo de trabajo de los requerimientos, 272
- flujo de trabajo de las pruebas, 291
- glosario, 273

- interfaz gráfica de usuario (GUI), 294
- modelo, 276
- modelo de negocio, 274
- negociar, 298
- observación directa, 275
- prototipo rápido, 293
- requerimiento funcional, 277
- requerimiento no funcional, 277
- restricción de la plataforma, 277
- tiempos de respuesta, 278

## Términos clave del estudio de caso

acuarela, 278  
calidad, 279  
coeficiente de moda, 285  
medio, 278  
naturaleza muerta, 278

obra maestra, 279  
óleo, 278  
otras pinturas, 279  
otros medios, 278  
paisaje, 278

parecido, 285  
retrato, 278  
tema, 278  
trabajo maestro, 279

## Problemas

- 10.1 Para el caso de estudio Osbert Oglesby, el diagrama de casos de uso del modelo de negocio inicial también es el diagrama de casos de uso del modelo de requerimientos inicial. ¿Será siempre esto cierto? Explique su respuesta.
- 10.2 Proporcione un requerimiento no funcional que pueda ser manejado sin tener el conocimiento detallado acerca del producto de software deseado.
- 10.3 Ahora, proporcione un requerimiento no funcional que tenga que ser manejado después de haber completado el flujo de trabajo de los requerimientos.
- 10.4 Diferencie entre un *caso de uso* y un *diagrama de casos de uso*.
- 10.5 Diferencie entre un *usuario* y un *actor*.
- 10.6 Dibuje un diagrama de flujo representando el flujo de trabajo de los requerimientos.
- 10.7 Acaba de unirse a Victoria & Niagara Software como gerente de software. Victoria & Niagara ha estado desarrollando software de contabilidad para pequeños negocios por muchos años usando el modelo de cascada, regularmente con éxito. Con base en su experiencia, piensa que el Proceso Unificado es una forma muy superior de desarrollar software. Escriba un informe dirigido al vicepresidente de desarrollo de software explicando por qué cree usted que la organización debe cambiar al Proceso Unificado. Recuerde que a los vicepresidentes no les gustan los informes de más de media página.
- 10.8 Usted es el vicepresidente de desarrollo de software de Victoria & Niagara. Conteste al informe del problema 10.7.
- 10.9 ¿Cuál es el resultado de no construir un prototipo rápido velocemente?
- 10.10 (Proyecto de análisis y diseño) Realice el flujo de trabajo de los requerimientos para el sistema de circulación de la biblioteca automatizada del problema 8.7.
- 10.11 (Proyecto de análisis y diseño) Realice el flujo de trabajo de los requerimientos del producto para determinar si el estado de cuenta del banco del problema 8.8 es correcto.
- 10.12 (Proyecto de análisis y diseño) Realice el flujo de trabajo de los requerimientos para cajero automático (ATM) del problema 8.9.
- 10.13 (Proyecto de término) Realice el flujo de trabajo de los requerimientos para el proyecto Ophelia's Oasis del apéndice A.
- 10.14 (Caso de estudio) Osbert Oglesby ha decidido expandir su negocio vendiendo a consignación. Esto es, él acepta pinturas para vender que colgará en su galería. Si la pintura se vende en tres meses al precio previamente acordado entre Osbert y el dueño de la pintura, entonces Osbert y el dueño comparten el precio de venta por partes iguales. De lo contrario, la pintura se le regresa al dueño y el dinero no cambia de manos. Trace el caso de uso para las ventas a consignación. Proporcione la descripción del caso de uso que dibujó para el problema 10.4. Provea el mayor detalle que pueda.
- 10.15 (Caso de estudio) Se debe generar un informe de las ventas a consignación (problema 10.14) durante el año pasado. Modifique las figuras 10.6 y 10.19 apropiadamente para incorporar este informe adicional.
- 10.16 (Caso de estudio) Utilice la información de las secciones 10.6 a 10.9, y construya un prototipo rápido para el caso de estudio Osbert Oglesby. Utilice el software y hardware que su instructor especifique.
- 10.17 (Lecturas sobre ingeniería de software) Su instructor le distribuirá copias de [Alexander, 2003]. ¿Está de acuerdo en que los casos de mal uso son importantes?

## Referencias



- [Alexander, 2003] I. ALEXANDER , “Misuse Cases: Use Cases with Hostile Intent”, *IEEE Software* **20** (enero/febrero 2003), pp. 58-66.
- [Berry, 2002] D. M. BERRY, “The Inevitable Pain of Software Development: Why There is No Silver Bullet”, Technical Report, University of Waterloo, School of Computer Science, Waterloo, Canadá, noviembre 2002.
- [Brooks, 1975] F. P. BROOKS, JR., *The Mythical Man-Month: Essays on Software Engineering*, Addison Wesley, Reading, MA, 1975; Twentieth Anniversary Edition, Addison Wesley, Reading, MA, 1995.
- [Browne, 1994] D. BROWNE, *STUDIO: STructured User-Interface Design for Interaction Optimization*, Prentice-Hall, Englewood Cliffs, NJ, 1994.
- [Connell y Shafer, 1989] J. L. CONNELL y L. SHAFER, *Structured Rapid Prototyping: An Evolutionary Approach to Software Development*, Yourdon Press, Englewood Cliffs, NJ, 1989.
- [Dix, Finlay, Abowd y Beale, 1993] A. DIX, J. FINLAY, G. ABOWD y R. BEALE, *Human Computer Interaction*, Prentice-Hall, Englewood Cliffs, NJ, 1993.
- [Gane, 1989] S. GANE, *Rapid System Development: Using Structured Techniques and Relational Technology*, Prentice-Hall, Englewood Cliffs, NJ, 1989.
- [Jackson, 1995] M. JACKSON, *Software Requirements and Specifications: A Lexicon of Practice, Principles and Prejudices*, Addison Wesley Longman, Reading, MA, 1995
- [Jacobson, Booch y Rumbaugh, 1999] I. JACOBSON, G. BOOCH y J. RUMBAUGH, *The Unified Software Development Process*, Addison Wesley, Reading, MA, 1999.
- [Karlsson y Ryan, 1997] J. KARLSSON y K. RYAN, “A Cost-Value Approach for Prioritizing Requirements”, *IEEE Software* **14** (septiembre/octubre 1997), pp. 67-74.
- [Preece, 1994] J. PREECE, *Human-Computer Interaction*, Addison Wesley, Reading, MA, 1994.
- [Schach y Wood, 1986] S. R. SCHACH y P. T. WOOD, “An Almost Path-Free Very High-Level Interactive Data Manipulation Language for a Microcomputer-Based Database System”, *Software—Practice and Experience* **16** (marzo 1986), pp. 243-268.
- [Shneiderman, 2003] B. SHNEIDERMAN, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 4a. ed., Addison Wesley Longman, Reading, MA, 2003.
- [Thayer y Dorfman, 1999] R. H. THAYER y M. DORFMAN, *Software Requirements Engineering*, 2a. ed. revisada, IEEE Computer Society Press, Los Alamitos, CA, 1999.

# Tema 3

El proceso de Ingeniería de  
Requerimientos



# 3 Comprensión de los requerimientos

Entender los requerimientos de un problema es una de las tareas más difíciles que enfrenta el ingeniero de software. Cuando se piensa por primera vez, no parece tan difícil desarrollar un entendimiento claro de los requerimientos. Después de todo, ¿acaso no sabe el cliente lo que se necesita? ¿No deberían tener los usuarios finales una buena comprensión de las características y funciones que le darán un beneficio? Sorprendentemente, en muchas instancias la respuesta a estas preguntas es “no”. E incluso si los clientes y los usuarios finales explican sus necesidades, éstas cambiarán mientras se desarrolla el proyecto.

En el prólogo a un libro escrito por Ralph Young [You01] sobre las prácticas eficaces respecto de los requerimientos, escribió lo siguiente:

Es la peor de las pesadillas. Un cliente entra a la oficina, toma asiento, lo mira a uno fijamente a los ojos y dice: “Sé que cree que entiende lo que digo, pero lo que usted no entiende es que lo que digo no es lo que quiero decir.” Invariablemente, esto pasa cuando ya está avanzado el proyecto, después de que se han hecho compromisos con los plazos de entrega, que hay reputaciones en juego y mucho dinero invertido.

Todos los que hemos trabajado en el negocio de los sistemas y del software durante algunos años hemos vivido la pesadilla descrita, pero pocos hemos aprendido a escapar. Batallamos cuando tratamos de obtener los requerimientos de nuestros clientes. Tenemos problemas para entender la información que obtenemos. Es frecuente que registremos los requerimientos de manera desorganizada y que dediquemos muy poco tiempo a verificar lo que registramos. Dejamos que el cambio nos controle en lugar de establecer mecanismos para controlarlo a él. En pocas palabras, fallamos en establecer un fundamento sólido para el sistema o software. Cada uno de los problemas es difícil. Cuando se combinan, el panorama es atemorizador aun para los gerentes y profesionales más experimentados. Pero hay solución.

## Una mirada rápida

*:Qué es?* Antes de comenzar cualquier trabajo técnico es una buena idea aplicar un conjunto de tareas de ingeniería a los requerimientos. Éstas llevarán a la comprensión de cuál será el efecto que tendrá el software en el negocio, qué es lo que quiere el cliente y cómo interactuarán los usuarios finales con el software.

## CONCEPTOS CLAVE

administración de los requerimientos .....	105
casos de uso .....	113
colaboración .....	107
concepción .....	102
despliegue de la función de calidad .....	111
elaboración .....	117
especificación .....	104
indagación .....	103
indagación de los requerimientos .....	108
ingeniería de requerimientos .....	102
modelo del análisis .....	117
negociación .....	121
participantes .....	106
patrones de análisis .....	120
productos del trabajo .....	112
puntos de vista .....	107
validación .....	105
validación de los requerimientos .....	122

**¿Quién lo hace?** Los ingenieros de software (que en el mundo de las tecnologías de información a veces son llamados *ingenieros de sistemas o analistas*) y todos los demás participantes del proyecto (gerentes, clientes y usuarios) intervienen en la ingeniería de requerimientos.

**¿Por qué es importante?** Diseñar y construir un elegante programa de cómputo que resuelva el problema equivocado no satisface las necesidades de nadie. Por eso es importante entender lo que el cliente desea antes de comenzar a diseñar y a construir un sistema basado en computadora.

**¿Cuáles son los pasos?** La ingeniería de requerimientos comienza con la concepción, tarea que define el alcance y la naturaleza del problema que se va a resolver. Va seguida de la indagación, labor que ayuda a los participantes a definir lo que se requiere. Después sigue la elaboración, donde se refinan y modifican los requerimientos básicos. Cuando los participantes definen el problema, tiene lugar una negociación: ¿cuáles son las prioridades, qué es lo esencial, cuándo se requiere? Por último, se especifica el problema de algún modo y luego se revisa o valida para garantizar que hay coincidencia entre la comprensión que usted tiene del problema y la que tienen los participantes.

**¿Cuál es el producto final?** El objetivo de los requerimientos de ingeniería es proporcionar a todas las partes un entendimiento escrito del problema. Esto se logra por medio de varios productos del trabajo: escenarios de uso, listas de funciones y de características, modelos de requerimientos o especificaciones.

**¿Cómo me aseguro de que lo hice bien?** Se revisan con los participantes los productos del trabajo de la ingeniería de requerimientos a fin de asegurar que lo que se aprendió es lo que ellos quieren decir en realidad. Aquí cabe una advertencia: las cosas cambiarán aun después de que todas las partes estén de acuerdo, y seguirán cambiando durante todo el proyecto.

Es razonable afirmar que las técnicas que se estudiarán en este capítulo no son una “solución” verdadera para los retos que se mencionaron, pero sí proveen de un enfoque sólido para enfrentarlos.

## 5.1 Ingeniería de requerimientos

El diseño y construcción de software de computadora es difícil, creativo y sencillamente divertido. En realidad, elaborar software es tan atractivo que muchos desarrolladores de software quieren ir directo a él antes de haber tenido el entendimiento claro de lo que se necesita. Argumentan que las cosas se aclararán a medida que lo elaboren, que los participantes en el proyecto podrán comprender sus necesidades sólo después de estudiar las primeras iteraciones del software, que las cosas cambian tan rápido que cualquier intento de entender los requerimientos en detalle es una pérdida de tiempo, que las utilidades salen de la producción de un programa que funcione y que todo lo demás es secundario. Lo que hace que estos argumentos sean tan seductores es que tienen algunos elementos de verdad.<sup>1</sup> Pero todos son erróneos y pueden llevar un proyecto de software al fracaso.

El espectro amplio de tareas y técnicas que llevan a entender los requerimientos se denomina *ingeniería de requerimientos*. Desde la perspectiva del proceso del software, la ingeniería de requerimientos es una de las acciones importantes de la ingeniería de software que comienza durante la actividad de comunicación y continúa en la de modelado. Debe adaptarse a las necesidades del proceso, del proyecto, del producto y de las personas que hacen el trabajo.

La ingeniería de requerimientos tiende un puente para el diseño y la construcción. Pero, ¿dónde se origina el puente? Podría argumentarse que principia en los pies de los participantes en el proyecto (por ejemplo, gerentes, clientes y usuarios), donde se definen las necesidades del negocio, se describen los escenarios de uso, se delinean las funciones y características y se identifican las restricciones del proyecto. Otros tal vez sugieran que empieza con una definición más amplia del sistema, donde el software no es más que un componente del dominio del sistema mayor. Pero sin importar el punto de arranque, el recorrido por el puente lo lleva a uno muy alto sobre el proyecto, lo que le permite examinar el contexto del trabajo de software que debe realizarse; las necesidades específicas que deben abordar el diseño y la construcción; las prioridades que guían el orden en el que se efectúa el trabajo, y la información, las funciones y los comportamientos que tendrán un profundo efecto en el diseño resultante.

La ingeniería de requerimientos proporciona el mecanismo apropiado para entender lo que desea el cliente, analizar las necesidades, evaluar la factibilidad, negociar una solución razonable, especificar la solución sin ambigüedades, validar la especificación y administrar los requerimientos a medida de que se transforman en un sistema funcional [Tha97]. Incluye siete tareas diferentes: concepción, indagación, elaboración, negociación, especificación, validación y administración. Es importante notar que algunas de estas tareas ocurren en paralelo y que todas se adaptan a las necesidades del proyecto.

**Concepción.** ¿Cómo inicia un proyecto de software? ¿Existe un solo evento que se convierte en el catalizador de un nuevo sistema o producto basado en computadora o la necesidad evoluciona en el tiempo? No hay respuestas definitivas a estas preguntas. En ciertos casos, una conversación casual es todo lo que se necesita para desencadenar un trabajo grande de ingeniería de software. Pero en general, la mayor parte de proyectos comienzan cuando se identifica una necesidad del negocio o se descubre un nuevo mercado o servicio potencial. Los participantes de la comunidad del negocio (por ejemplo, los directivos, personal de mercadotecnia, gerentes de producto, etc.) definen un caso de negocios para la idea, tratan de identificar el ritmo y profundidad del mercado, hacen un análisis de gran visión de la factibilidad e identifican una descripción funcional del alcance del proyecto. Toda esta información está sujeta a cambio, pero es suficiente para desencadenar análisis con la organización de ingeniería de software.<sup>2</sup>

**cita:** "La parte más difícil al construir un sistema de software es decidir qué construir. Ninguna parte del trabajo invalida tanto al sistema resultante si ésta se hace mal. Nada es más difícil de corregir después."

Fred Brooks

**punto clave** La ingeniería de requerimientos establece una base sólida para el diseño y la construcción. Sin ésta, el software resultante tiene alta probabilidad de no satisfacer las necesidades del cliente.

**consejo** Espere hacer un poco de diseño al recabar los requerimientos, y un poco de requerimientos durante el trabajo de diseño.

**cita:** "Las semillas de los desastres enormes del software por lo general se vislumbran en los tres primeros meses del inicio del proyecto."

Coper Jones

1 Esto es cierto en particular para los proyectos pequeños (menos de un mes) y muy pequeños, que requieren relativamente poco esfuerzo de software sencillo. A medida que el software crece en tamaño y complejidad, estos argumentos comienzan a ser falsos.

2 Si va a desarrollarse un sistema basado en computadora, los análisis comienzan en el contexto de un proceso de ingeniería de sistemas. Para más detalles de la ingeniería de sistemas, visite el sitio web de esta obra.

En la concepción del proyecto,<sup>3</sup> se establece el entendimiento básico del problema, las personas que quieren una solución, la naturaleza de la solución que se desea, así como la eficacia de la comunicación y colaboración preliminares entre los otros participantes y el equipo de software.



¿Por qué es difícil llegar al entendimiento claro de lo que quiere el cliente?

**Indagación.** En verdad que parece muy simple: preguntar al cliente, a los usuarios y a otras personas cuáles son los objetivos para el sistema o producto, qué es lo que va a lograrse, cómo se ajusta el sistema o producto a las necesidades del negocio y, finalmente, cómo va a usarse el sistema o producto en las operaciones cotidianas. Pero no es simple: es muy difícil.

Christel y Kang [Cri92] identificaron cierto número de problemas que se encuentran cuando ocurre la indagación:

- **Problemas de alcance.** La frontera de los sistemas está mal definida o los clientes o usuarios finales especifican detalles técnicos innecesarios que confunden, más que clarifican, los objetivos generales del sistema.
- **Problemas de entendimiento.** Los clientes o usuarios no están completamente seguros de lo que se necesita, comprenden mal las capacidades y limitaciones de su ambiente de computación, no entienden todo el dominio del problema, tienen problemas para comunicar las necesidades al ingeniero de sistemas, omiten información que creen que es “obvia”, especifican requerimientos que están en conflicto con las necesidades de otros clientes o usuarios, o solicitan requerimientos ambiguos o que no pueden someterse a prueba.
- **Problemas de volatilidad.** Los requerimientos cambian con el tiempo.

Para superar estos problemas, debe enfocarse la obtención de requerimientos en forma organizada.

**consejo** *La elaboración es algo bueno, pero hay que saber cuándo detenerse. La clave es describir el problema en forma que establezca una base firme para el diseño. Si se trabaja más allá de este punto, se está haciendo diseño.*

**consejo** *En una negociación eficaz no debe haber ganador ni perdedor. Ambos lados ganan porque un “trato” con el que ambas partes pueden vivir es algo sólido.*

**punto clave** La formalidad y el formato de una especificación varían con el tamaño y complejidad del software que se va a construir.

**Elaboración.** La información obtenida del cliente durante la concepción e indagación se expande y refina durante la elaboración. Esta tarea se centra en desarrollar un modelo refinado de los requerimientos (véanse los capítulos 6 y 7) que identifique distintos aspectos de la función del software, su comportamiento e información.

La elaboración está motivada por la creación y mejora de escenarios de usuario que describan cómo interactuará el usuario final (y otros actores) con el sistema. Cada escenario de usuario se enumera con sintaxis apropiada para extraer clases de análisis, que son entidades del dominio del negocio visibles para el usuario final. Se definen los atributos de cada clase de análisis y se identifican los servicios<sup>4</sup> que requiere cada una de ellas. Se identifican las relaciones y colaboración entre clases, y se producen varios diagramas adicionales.

**Negociación.** No es raro que los clientes y usuarios pidan más de lo que puede lograrse dado lo limitado de los recursos del negocio. También es relativamente común que distintos clientes o usuarios propongan requerimientos conflictivos con el argumento de que su versión es “esencial para nuestras necesidades especiales”.

Estos conflictos deben reconciliarse por medio de un proceso de negociación. Se pide a clientes, usuarios y otros participantes que ordenen sus requerimientos según su prioridad y que después analicen los conflictos. Con el empleo de un enfoque iterativo que da prioridad a los requerimientos, se evalúa su costo y riesgo, y se enfrentan los conflictos internos; algunos requerimientos se eliminan, se combinan o se modifican de modo que cada parte logre cierto grado de satisfacción.

**Especificación.** En el contexto de los sistemas basados en computadora (y software), el término *especificación* tiene diferentes significados para distintas personas. Una especificación puede ser un documento escrito, un conjunto de modelos gráficos, un modelo matemático formal, un conjunto de escenarios de uso, un prototipo o cualquier combinación de éstos.

<sup>3</sup> Recuerde que el proceso unificado (véase el capítulo 2) define una “fase de concepción” más amplia que incluye las fases de concepción, indagación y elaboración, que son estudiadas en dicho capítulo.

<sup>4</sup> Un servicio manipula los datos agrupados por clase. También se utilizan los términos *operación* y *método*. Si no está familiarizado con conceptos de la orientación a objetos, consulte el apéndice 2, en el que se presenta una introducción básica.

Algunos sugieren que para una especificación debe desarrollarse y utilizarse una “plantilla estándar” [Som97], con el argumento de que esto conduce a requerimientos presentados en forma consistente y por ello más comprensible. Sin embargo, en ocasiones es necesario ser flexible cuando se desarrolla una especificación. Para sistemas grandes, el mejor enfoque puede ser un documento escrito que combine descripciones en un lenguaje natural con modelos gráficos. No obstante, para productos o sistemas pequeños que residan en ambientes bien entendidos, quizá todo lo que se requiera sea escenarios de uso.



## Formato de especificación de requerimientos de software

Una *especificación de requerimientos de software* (ERS) es un documento que se crea cuando debe especificarse una descripción detallada de todos los aspectos del software que se va a elaborar, antes de que el proyecto comience. Es importante notar que una ERS formal no siempre está en forma escrita. En realidad, hay muchas circunstancias en las que el esfuerzo dedicado a la ERS estaría mejor aprovechado en otras actividades de la ingeniería de software. Sin embargo, se justifica la ERS cuando el software va a ser desarrollado por una tercera parte, cuando la falta de una especificación crearía problemas severos al negocio, si un sistema es complejo en extremo o si se trata de un negocio de importancia crítica.

Karl Wiegert [Wie03], de la empresa Process Impact Inc., desarrolló un formato útil (disponible en [www.processimpact.com/process\\_assets/srs\\_template.doc](http://www.processimpact.com/process_assets/srs_template.doc)) que sirve como guía para aquellos que deben crear una ERS completa. Su contenido normal es el siguiente:

### Tabla de contenido Revisión de la historia

#### 1. Introducción

- 1.1 Propósito
- 1.2 Convenciones del documento
- 1.3 Audiencia objetivo y sugerencias de lectura
- 1.4 Alcance del proyecto
- 1.5 Referencias

#### 2. Descripción general

- 2.1 Perspectiva del producto
- 2.2 Características del producto
- 2.3 Clases y características del usuario
- 2.4 Ambiente de operación
- 2.5 Restricciones de diseño e implementación
- 2.6 Documentación para el usuario
- 2.7 Suposiciones y dependencias

#### 3. Características del sistema

- 3.1 Característica 1 del sistema
- 3.2 Característica 2 del sistema (y así sucesivamente)

#### 4. Requerimientos de la interfaz externa

- 4.1 Interfaces de usuario
- 4.2 Interfaces del hardware
- 4.3 Interfaces del software
- 4.4 Interfaces de las comunicaciones

#### 5. Otros requerimientos no funcionales

- 5.1 Requerimientos de desempeño
- 5.2 Requerimientos de seguridad
- 5.3 Requerimientos de estabilidad
- 5.4 Atributos de calidad del software

#### 6. Otros requerimientos

- Apéndice A: Glosario**
- Apéndice B: Modelos de análisis**
- Apéndice C: Lista de conceptos**

Puede obtenerse una descripción detallada de cada ERS si se descarga el formato desde la URL mencionada antes.

Información

**consejo** Un aspecto clave durante la validación de los requerimientos es la consistencia. Utilice el modelo de análisis para asegurar que los requerimientos se han enunciado de manera consistente.

## El proceso de ingeniería de requerimientos

**Validación.** La calidad de los productos del trabajo que se generan como consecuencia de la ingeniería de los requerimientos se evalúa durante el paso de validación. La validación de los requerimientos analiza la especificación<sup>5</sup> a fin de garantizar que todos ellos han sido enunciados sin ambigüedades; que se detectaron y corrigieron las inconsistencias, las omisiones y los errores, y que los productos del trabajo se presentan conforme a los estándares establecidos para el proceso, el proyecto y el producto.

El mecanismo principal de validación de los requerimientos es la revisión técnica (véase el capítulo 15). El equipo de revisión que los valida incluye ingenieros de software, clientes, usuarios y otros participantes, que analizan la especificación en busca de errores de contenido o de interpretación, de aspectos en los que tal vez se requiera hacer aclaraciones, falta de información, inconsistencias (problema notable cuando se hace la ingeniería de productos o sistemas grandes) y requerimientos en conflicto o irreales (no asequibles).

# Información



### Lista de verificación para validar requerimientos

Con frecuencia es útil analizar cada requerimiento en comparación con preguntas de verificación. A continuación se presentan algunas:

- ¿Los requerimientos están enunciados con claridad? ¿Podrían interpretarse mal?
- ¿Está identificada la fuente del requerimiento (por ejemplo, una persona, reglamento o documento)? ¿Se ha estudiado el planteamiento final del requerimiento en comparación con la fuente original?
- ¿El requerimiento está acotado en términos cuantitativos?
- ¿Qué otros requerimientos se relacionan con éste? ¿Están comparados con claridad por medio de una matriz de referencia cruzada u otro mecanismo?
- ¿El requerimiento viola algunas restricciones del dominio?
- ¿Puede someterse a prueba el requerimiento? Si es así, ¿es posible especificar las pruebas (en ocasiones se denominan criterios de validación) para ensayar el requerimiento?
- ¿Puede rastrearse el requerimiento hasta cualquier modelo del sistema que se haya creado?
- ¿Es posible seguir el requerimiento hasta los objetivos del sistema o producto?
- ¿La especificación está estructurada en forma que lleva a entenderlo con facilidad, con referencias y traducción fáciles a productos del trabajo más técnicos?
- ¿Se ha creado un índice para la especificación?
- ¿Están enunciadas con claridad las asociaciones de los requerimientos con las características de rendimiento, comportamiento y operación? ¿Cuáles requerimientos parecen ser implícitos?

<sup>5</sup> Recuerde que la naturaleza de la especificación variará con cada proyecto. En ciertos casos, la “especificación” no es más que un conjunto de escenarios de usuario. En otros, la especificación tal vez sea un documento que contiene escenarios, modelos y descripciones escritas.

**Administración de los requerimientos.** Los requerimientos para sistemas basados en computadora cambian, y el deseo de modificarlos persiste durante toda la vida del sistema. La administración de los requerimientos es el conjunto de actividades que ayudan al equipo del proyecto a identificar, controlar y dar seguimiento a los requerimientos y a sus cambios en cualquier momento del desarrollo del proyecto.<sup>6</sup> Muchas de estas actividades son idénticas a las técnicas de administración de la configuración del software (TAS) que se estudian en el capítulo 22.



## Ingeniería de requerimientos

**Objetivo:** Las herramientas de la ingeniería de los requerimientos ayudan a reunir éstos, a modelarlos, administrarlos y validarlos.

**Mecánica:** La mecánica de las herramientas varía. En general, éstas elaboran varios modelos gráficos (por ejemplo, UML) que ilustran los aspectos de información, función y comportamiento de un sistema. Estos modelos constituyen la base de todas las demás actividades del proceso de software.

### Herramientas representativas:<sup>7</sup>

En el sitio de Volere Requirements, en [www.volere.co.uk/tools.htm](http://www.volere.co.uk/tools.htm), se encuentra una lista razonablemente amplia (y actualizada) de herramientas para la ingeniería de requerimientos. En los capítulos 6 y 7 se estudian las herramientas que sirven para modelar aquéllos. Las que se mencionan a continuación se centran en su administración.

EasyRM, desarrollada por Cybernetic Intelligence GmbH ([www.easy-rm.com](http://www.easy-rm.com)), construye un diccionario/glosario especial para proyectos, que contiene descripciones y atributos detallados de los requerimientos.

Rational RequisitePro, elaborada por Rational Software ([www-306.ibm.com/software/awdtools/repro/](http://www-306.ibm.com/software/awdtools/repro/)), permite a los usuarios construir una base de datos de requerimientos, representar relaciones entre ellos y organizarlos, indicar su prioridad y rastrearlos.

En el sitio de Volere ya mencionado, se encuentran muchas herramientas adicionales para administrar requerimientos, así como en la dirección [www.jiludwig.com/Requirements\\_Management\\_Tools.html](http://www.jiludwig.com/Requirements_Management_Tools.html)

# Herramientas de software

## 5.2 Establecer las bases

En el caso ideal, los participantes e ingenieros de software trabajan juntos en el mismo equipo.<sup>8</sup> En esas condiciones, la ingeniería de requerimientos tan sólo consiste en sostener conversaciones significativas con colegas que sean miembros bien conocidos del equipo. Pero es frecuente que en la realidad esto sea muy diferente.

Los clientes o usuarios finales tal vez se encuentren en ciudades o países diferentes, quizás sólo tengan una idea vaga de lo que se requiere, puede ser que tengan opiniones en conflicto sobre el

6 La administración formal de los requerimientos sólo se practica para proyectos grandes que tienen cientos de requerimientos identificables. Para proyectos pequeños, esta actividad tiene considerablemente menos formalidad.

7 Las herramientas mencionadas aquí no son obligatorias sino una muestra de las que hay en esta categoría. En la mayoría de casos, los nombres de las herramientas son marcas registradas por sus respectivos desarrolladores.

8 Este enfoque es ampliamente recomendable para proyectos que adoptan la filosofía de desarrollo de software ágil.

sistema que se va a elaborar, que posean un conocimiento técnico limitado o que dispongan de poco tiempo para interactuar con el ingeniero que recabará los requerimientos. Ninguna de estas posibilidades es deseable, pero todas son muy comunes y es frecuente verse forzado a trabajar con las restricciones impuestas por esta situación.

En las secciones que siguen se estudian las etapas requeridas para establecer las bases que permiten entender los requerimientos de software a fin de que el proyecto comience en forma tal que se mantenga avanzando hacia una solución exitosa.

**punto clave** Un *participante* es cualquier persona que tenga interés directo o que se beneficie del sistema que se va a desarrollar.

**cita:** "Ponga a tres participantes en un cuarto y pregúnteleles qué clase de sistema quieren. Es probable que escuche cuatro o más opiniones diferentes."

Anónimo

### > 5.2.1 Identificación de los participantes

Sommerville y Sawyer [Som97] definen *participante* como "cualquier persona que se beneficie en forma directa o indirecta del sistema en desarrollo". Ya se identificaron los candidatos habituales: gerentes de operaciones del negocio, gerentes de producto, personal de mercadotecnia, clientes internos y externos, usuarios finales, consultores, ingenieros de producto, ingenieros de software e ingenieros de apoyo y mantenimiento, entre otros. Cada participante tiene un punto de vista diferente respecto del sistema, obtiene distintos beneficios cuando éste se desarrolla con éxito y corre distintos riesgos si fracasa el esfuerzo de construcción.

Durante la concepción, debe hacerse la lista de personas que harán aportes cuando se recaben los requerimientos (véase la sección 5.3). La lista inicial crecerá cuando se haga contacto con los participantes porque a cada uno se le hará la pregunta: "¿A quién más piensa que debe consultarse?"

### > 5.2.2 Reconocer los múltiples puntos de vista

Debido a que existen muchos participantes distintos, los requerimientos del sistema se explorarán desde muchos puntos de vista diferentes. Por ejemplo, el grupo de mercadotecnia se interesa en funciones y características que estimularán el mercado potencial, lo que hará que el nuevo sistema sea fácil de vender. Los gerentes del negocio tienen interés en un conjunto de características para que se elabore dentro del presupuesto y que esté listo para ocupar nichos de mercado definidos. Los usuarios finales tal vez quieran características que les resulten familiares y que sean fáciles de aprender y usar. Los ingenieros de software quizás piensen en funciones invisibles para los participantes sin formación técnica, pero que permitan una infraestructura que dé apoyo a funciones y características más vendibles. Los ingenieros de apoyo tal vez se centren en la facilidad del software para recibir mantenimiento.

Cada uno de estos integrantes (y otros más) aportará información al proceso de ingeniería de los requerimientos. A medida que se recaba información procedente de múltiples puntos de vista, los requerimientos que surjan tal vez sean inconsistentes o estén en conflicto uno con otro. Debe clasificarse toda la información de los participantes (incluso los requerimientos inconsistentes y conflictivos) en forma que permita a quienes toman las decisiones escoger para el sistema un conjunto de requerimientos que tenga coherencia interna.

### > 5.2.3 Trabajar hacia la colaboración

Si en un proyecto de software hay involucrados cinco participantes, tal vez se tengan cinco (o más) diferentes opiniones acerca del conjunto apropiado de requerimientos. En los primeros capítulos se mencionó que, para obtener un sistema exitoso, los clientes (y otros participantes) debían colaborar entre sí (sin pelear por insignificancias) y con los profesionales de la ingeniería de software. Pero, ¿cómo se llega a esta colaboración?

El trabajo del ingeniero de requerimientos es identificar las áreas de interés común (por ejemplo, requerimientos en los que todos los participantes estén de acuerdo) y las de conflicto o incongruencia (por ejemplo, requerimientos que desea un participante, pero que están en conflicto con las necesidades de otro). Es la última categoría la que, por supuesto, representa un reto.



## Uso de “puntos de prioridad”

Una manera de resolver requerimientos conflictivos y, al mismo tiempo, mejorar la comprensión de la importancia relativa de todos, es usar un esquema de “votación” con base en *puntos de prioridad*. Se da a todos los participantes cierto número de puntos de prioridad que pueden “gastarse” en cualquier número de requerimientos. Se presenta una lista de éstos y cada participante indica la importancia relativa de cada uno (desde su punto de vista) con la asignación de uno o más puntos de prioridad. Los puntos gastados ya no pueden utilizarse otra vez. Cuando un participante agota sus puntos de prioridad, ya no tiene la posibilidad de hacer algo con los requerimientos. El total de puntos asignados a cada requerimiento por los participantes da una indicación de la importancia general de cada requerimiento.

Información

La colaboración no significa necesariamente que todos los requerimientos los defina un comité. En muchos casos, los participantes colaboran con la aportación de su punto de vista respecto de los requerimientos, pero un influyente “campeón del proyecto” (por ejemplo, el director del negocio o un tecnólogo experimentado) toma la decisión final sobre los requerimientos que lo integrarán.

### 5.2.4 Hacer las primeras preguntas

Las preguntas que se hacen en la concepción del proyecto deben estar “libres del contexto” [Gau89]. El primer conjunto de ellas se centran en el cliente y en otros participantes, en las metas y beneficios generales. Por ejemplo, tal vez se pregunte:

- ¿Quién está detrás de la solicitud de este trabajo?
- ¿Quién usará la solución?
- ¿Cuál será el beneficio económico de una solución exitosa?
- ¿Hay otro origen para la solución que se necesita?

**cita:** “Es mejor conocer algunas preguntas que todas las respuestas.”

James Thurber

Estas preguntas ayudan a identificar a todos los participantes con interés en el software que se va a elaborar. Además, las preguntas identifican el beneficio mensurable de una implementación exitosa y las posibles alternativas para el desarrollo de software personalizado.

Las preguntas siguientes permiten entender mejor el problema y hacen que el cliente exprese sus percepciones respecto de la solución:

- ¿Cuál sería una “buena” salida generada por una solución exitosa?
- ¿Qué problemas resolvería esta solución?
- ¿Puede mostrar (o describir) el ambiente de negocios en el que se usaría la solución?
- ¿Hay aspectos especiales del desempeño o restricciones que afecten el modo en el que se enfoca la solución?

¿Cuáles preguntas ayudarían a tener un entendimiento preliminar del problema?

Las preguntas finales se centran en la eficacia de la actividad de comunicación en sí. Gause y Weinberg [Gau89] las llaman “metapreguntas” y proponen la siguiente lista (abreviada):

- ¿Es usted la persona indicada para responder estas preguntas? ¿Sus respuestas son “oficiales”?
- ¿Mis preguntas son relevantes para el problema que se tiene?
- ¿Estoy haciendo demasiadas preguntas?
- ¿Puede otra persona dar información adicional?
- ¿Debería yo preguntarle algo más?

**cita:** “El que hace una pregunta es tonto durante cinco minutos; el que no la hace será tonto para siempre..”

Proverbio chino

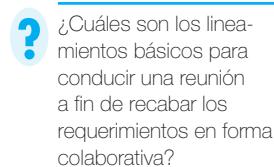
Estas preguntas (y otras) ayudarán a “romper el hielo” y a iniciar la comunicación, que es esencial para una indagación exitosa. Pero una reunión de preguntas y respuestas no es un enfoque que haya tenido un éxito apabullante. En realidad, la sesión de preguntas y respuestas sólo debe usarse para el primer encuentro y luego ser reemplazada por un formato de indagación de requerimientos que combine elementos de solución de problemas, negociación y especificación. En la sección 5.3 se presenta un enfoque de este tipo.

## 5.3 Indagación de los requerimientos

La indagación de los requerimientos (actividad también llamada *recabación de los requerimientos*) combina elementos de la solución de problemas, elaboración, negociación y especificación. A fin de estimular un enfoque colaborativo y orientado al equipo, los participantes trabajan juntos para identificar el problema, proponer elementos de la solución, negociar distintas visiones y especificar un conjunto preliminar de requerimientos para la solución [Zah90].<sup>9</sup>

### > 5.3.1 Recabación de los requerimientos en forma colaborativa

Se han propuesto muchos enfoques distintos para recabar los requerimientos en forma colaborativa. Cada uno utiliza un escenario un poco diferente, pero todos son variantes de los siguientes lineamientos básicos:



**cita:** “Dedicamos mucho tiempo —la mayor parte de todo el esfuerzo del proyecto— no a implementar o hacer pruebas, sino a tratar de decidir qué construir.”

Brian Lawrence

**WebRef** La solicitud conjunta de desarrollo (SCD) es una técnica popular para recabar requerimientos. En la dirección [www.carolla.com/wp-jad.htm](http://www.carolla.com/wp-jad.htm) se encuentra una buena descripción de ella.

- Tanto ingenieros de software como otros participantes dirigen o intervienen en las reuniones.
- Se establecen reglas para la preparación y participación.
- Se sugiere una agenda con suficiente formalidad para cubrir todos los puntos importantes, pero con la suficiente informalidad para que estimule el libre flujo de ideas.
- Un “facilitador” (cliente, desarrollador o participante externo) controla la reunión.
- Se utiliza un “mecanismo de definición” (que pueden ser hojas de trabajo, tablas sueltas, etiquetas adhesivas, pizarrón electrónico, grupos de conversación o foro virtual).

La meta es identificar el problema, proponer elementos de la solución, negociar distintos enfoques y especificar un conjunto preliminar de requerimientos de la solución en una atmósfera que favorezca el logro de la meta. Para entender mejor el flujo de eventos conforme ocurren, se presenta un escenario breve que bosqueja la secuencia de hechos que llevan a la reunión para obtener requerimientos, a lo que sucede durante ésta y a lo que sigue después de ella.

Durante la concepción (véase la sección 5.2), hay preguntas y respuestas básicas que establecen el alcance del problema y la percepción general de lo que constituye una solución. Fuera de estas reuniones iniciales, el desarrollador y los clientes escriben una o dos páginas de “solicitud de producto”.

Se selecciona un lugar, fecha y hora para la reunión, se escoge un facilitador y se invita a asistir a integrantes del equipo de software y de otras organizaciones participantes. Antes de la fecha de la reunión, se distribuye la solicitud de producto a todos los asistentes.

Por ejemplo,<sup>10</sup> considere un extracto de una solicitud de producto escrita por una persona de mercadotecnia involucrada en el proyecto *CasaSegura*. Esta persona escribe la siguiente narración sobre la función de seguridad en el hogar que va a ser parte de *CasaSegura*:

Nuestras investigaciones indican que el mercado para los sistemas de administración del hogar crece a razón de 40% anual. La primera función de *CasaSegura* que llevemos al mercado deberá ser la de seguridad del hogar. La mayoría de la gente está familiarizada con “sistemas de alarma”, por lo que ésta deberá ser fácil de vender.

La función de seguridad del hogar protegería, o reconocería, varias “situaciones” indeseables, como acceso ilegal, incendio y niveles de monóxido de carbono, entre otros. Emplearía sensores ina-

9 En ocasiones se denomina a este enfoque *técnica facilitada de especificación de la aplicación* (TSEA).

10 Este ejemplo (con extensiones y variantes) se usa para ilustrar métodos importantes de la ingeniería de software en muchos de los capítulos siguientes. Como ejercicio, sería provechoso que el lector realizará su propia reunión para recabar requerimientos y que desarrollara un conjunto de listas para ella.

lámbicos para detectar cada situación. Sería programada por el propietario y telefonearía en forma automática a una agencia de vigilancia cuando detectara una situación como las descritas.

En realidad, durante la reunión para recabar los requerimientos, otros contribuirían a esta narración y se dispondría de mucha más información. Pero aun con ésta habría ambigüedad, sería probable que existieran omisiones y ocurrieran errores. Por ahora bastará la “descripción funcional” anterior.

Mientras se revisa la solicitud del producto antes de la reunión, se pide a cada asistente que elabore una lista de objetos que sean parte del ambiente que rodeará al sistema, los objetos que producirá éste y los que usará para realizar sus funciones. Además, se solicita a cada asistente que haga otra lista de servicios (procesos o funciones) que manipulen o interactúen con los objetos. Por último, también se desarrollan listas de restricciones (por ejemplo, costo, tamaño, reglas del negocio, etc.) y criterios de desempeño (como velocidad y exactitud). Se informa a los asistentes que no se espera que las listas sean exhaustivas, pero sí que reflejen la percepción que cada persona tiene del sistema.

Entre los objetos descritos por *CasaSegura* tal vez estén incluidos el panel de control, detectores de humo, sensores en ventanas y puertas, detectores de movimiento, alarma, un evento (activación de un sensor), una pantalla, una computadora, números telefónicos, una llamada telefónica, etc. La lista de servicios puede incluir *configurar* el sistema, *preparar* la alarma, *vigilar* los sensores, *marcar* el teléfono, *programar* el panel de control y *leer* la pantalla (observe que los servicios actúan sobre los objetos). En forma similar, cada asistente desarrollará una lista de restricciones (por ejemplo, el sistema debe reconocer cuando los sensores no estén operando, debe ser amistoso con el usuario, debe tener una interfaz directa con una línea telefónica estándar, etc.) y de criterios de desempeño (un evento en un sensor debe reconocerse antes de un segundo, debe implementarse un esquema de prioridad de eventos, etcétera).

Las listas de objetos pueden adherirse a las paredes del cuarto con el empleo de pliegos de papel grandes o con láminas adhesivas, o escribirse en un tablero. Alternativamente, las listas podrían plasmarse en un boletín electrónico, sitio web interno o en un ambiente de grupo de conversación para revisarlas antes de la reunión. Lo ideal es que cada entrada de las listas pueda manipularse por separado a fin de combinar las listas o modificar las entradas y agregar otras. En esta etapa, están estrictamente prohibidos las críticas y el debate.

Una vez que se presentan las listas individuales acerca de un área temática, el grupo crea una lista, eliminando las entradas redundantes o agregando ideas nuevas que surjan durante el análisis, pero no se elimina ninguna. Despues de crear listas combinadas para todas las áreas temáticas, sigue el análisis, coordinado por el facilitador. La lista combinada se acorta, se alarga o se modifica su redacción para que refleje de manera apropiada al producto o sistema que se va a desarrollar. El objetivo es llegar a un consenso sobre la lista de objetos, servicios, restricciones y desempeño del sistema que se va a construir.

En muchos casos, un objeto o servicio descrito en la lista requerirá mayores explicaciones. Para lograr esto, los participantes desarrollan *miniespecificaciones* para las entradas en las listas.<sup>11</sup> Cada miniespecificación es una elaboración de un objeto o servicio. Por ejemplo, la correspondiente al objeto **Panel de control** de *CasaSegura* sería así:

El panel de control es una unidad montada en un muro, sus dimensiones aproximadas son de 9 por 5 pulgadas. Tiene conectividad inalámbrica con los sensores y con una PC. La interacción con el usuario tiene lugar por medio de un tablero que contiene 12 teclas. Una pantalla de cristal líquido de 3 por 3 pulgadas brinda retroalimentación al usuario. El software hace anuncios interactivos, como eco y funciones similares.

Las miniespecificaciones se presentan a todos los participantes para que sean analizadas. Se hacen adiciones, eliminaciones y otras modificaciones. En ciertos casos, el desarrollo de las miniespecificaciones descubrirá nuevos objetos, servicios o restricciones, o requerimientos de desempeño que se agregarán a las listas originales. Durante todos los análisis, el equipo debe posponer los aspectos que no puedan resolverse en la reunión. Se conserva una *lista de aspectos* para volver después a dichas ideas.

**consejo** Si un sistema o producto servirá a muchos usuarios, asegúrese de que los requerimientos se obtengan de una franja representativa de ellos. Si sólo uno define todos los requerimientos, el riesgo de no aceptación es elevado.

**cita:** “Los hechos no dejan de existir porque se les ignore.”

Aldous Huxley

**consejo** Evite el impulso de desechar alguna idea de un cliente con expresiones como “demasiado costosa” o “impráctica”. La intención aquí es negociar una lista aceptable para todos. Para lograrlo, debe tenerse la mente abierta.

<sup>11</sup> En vez de crear una miniespecificación, muchos equipos de software eligen desarrollar escenarios del usuario llamados *casos de uso*. Éstos se estudian en detalle en la sección 5.4 y en el capítulo 6.



## Conducción de una reunión para recabar los requerimientos

**La escena:** Sala de juntas. Está en marcha la primera reunión para recabar los requerimientos.

**Participantes:** Jamie Lazar, integrante del equipo de software; Vinod Raman, miembro del equipo de software; Ed Robbins, miembro del equipo de software; Doug Miller, gerente de ingeniería de software; tres trabajadores de mercadotecnia; un representante de ingeniería del producto, y un facilitador.

**La conversación:**

**Facilitador (apunta en un pizarrón):** De modo que ésa es la lista actual de objetos y servicios para la función de seguridad del hogar.

**Persona de mercadotecnia:** Eso la cubre, desde nuestro punto de vista.

**Vinod:** ¿No dijo alguien que quería que toda la funcionalidad de CasaSegura fuera accesible desde internet? Eso incluiría la función de seguridad, ¿o no?

**Persona de mercadotecnia:** Sí, así es... tendremos que añadir esa funcionalidad y los objetos apropiados.

**Facilitador:** ¿Agrega eso algunas restricciones?

**Jamie:** Sí, tanto técnicas como legales.

**Representante del producto:** ¿Qué significa eso?

**Jamie:** Nos tendríamos que asegurar de que un extraño no pueda ingresar al sistema, desactivarlo y robar en el lugar o hacer algo peor. Mucha responsabilidad sobre nosotros.

**Doug:** Muy cierto.

**Mercadotecnia:** Pero lo necesitamos así... sólo asegúrense de impedir que ingrese un extraño.

**Ed:** Eso es más fácil de decir que de hacer.

**Facilitador (interrumpe):** No quiero que debatamos esto ahora. Anotémoslo como un aspecto y continuemos.

(Doug, que es el secretario de la reunión, toma debida nota.)

**Facilitador:** Tengo la sensación de que hay más por considerar aquí.

(El grupo dedica los siguientes 20 minutos a mejorar y aumentar los detalles de la función de seguridad del hogar.)

**punto clave** El DFC define los requerimientos de forma que maximicen la satisfacción del cliente.

**consejo** Todos desean implementar muchos requerimientos emocionantes, pero hay que tener cuidado. Así es como empiezan a "quedarse lisos los requerimientos". Pero en contrapartida, los requerimientos emocionantes llevan a un avance enorme del producto...

### > 5.3.2 Despliegue de la función de calidad

El *despliegue de la función de calidad* (DFC) es una técnica de administración de la calidad que traduce las necesidades del cliente en requerimientos técnicos para el software. El DFC "se concentra en maximizar la satisfacción del cliente a partir del proceso de ingeniería del software" [Zul92]. Para lograr esto, el DFC pone el énfasis en entender lo que resulta valioso para el cliente y luego despliega dichos valores en todo el proceso de ingeniería. El DFC identifica tres tipos de requerimientos [Zul92]:

**Requerimientos normales.** Objetivos y metas que se establecen para un producto o sistema durante las reuniones con el cliente. Si estos requerimientos están presentes, el cliente queda satisfecho. Ejemplos de requerimientos normales son los tipos de gráficos pedidos para aparecer en la pantalla, funciones específicas del sistema y niveles de rendimiento definidos.

**Requerimientos esperados.** Están implícitos en el producto o sistema y quizás sean tan importantes que el cliente no los mencione de manera explícita. Su ausencia causará mucha insa-

tisfacción. Algunos ejemplos de requerimientos esperados son: fácil interacción humano/máquina, operación general correcta y confiable, y facilidad para instalar el software.

**Requerimientos emocionantes.** Estas características van más allá de las expectativas del cliente y son muy satisfactorias si están presentes. Por ejemplo, el software para un nuevo teléfono móvil viene con características estándar, pero si incluye capacidades inesperadas (como pantalla sensible al tacto, correo de voz visual, etc.) agrada a todos los usuarios del producto.

Aunque los conceptos del DFC son aplicables en todo el proceso del software [Par96a], hay técnicas específicas de aquél que pueden aplicarse a la actividad de indagación de los requerimientos. El DFC utiliza entrevistas con los clientes, observación, encuestas y estudio de datos históricos (por ejemplo, reportes de problemas) como materia prima para la actividad de recabación de los requerimientos. Después, estos datos se llevan a una tabla de requerimientos —llamada *tabla de la voz del cliente*— que se revisa con el cliente y con otros participantes. Luego se emplean varios diagramas, matrices y métodos de evaluación para extraer los requerimientos esperados y tratar de percibir requerimientos emocionantes [Aka04].

**WebRef** En la dirección [www.qfdi.org](http://www.qfdi.org) se encuentra información útil sobre el DFC.

### > 5.3.3 Escenarios de uso

A medida que se reúnen los requerimientos, comienza a materializarse la visión general de funciones y características del sistema. Sin embargo, es difícil avanzar hacia actividades más técnicas de la ingeniería de software hasta no entender cómo emplearán los usuarios finales dichas funciones y características. Para lograr esto, los desarrolladores y usuarios crean un conjunto de escenarios que identifican la naturaleza de los usos para el sistema que se va a construir. Los escenarios, que a menudo se llaman *casos de uso* [Jac92], proporcionan la descripción de la manera en la que se utilizará el sistema. Los casos de uso se estudian con más detalle en la sección 5.4.



#### Desarrollo de un escenario preliminar de uso

CasaSegura

**La escena:** Una sala de juntas, donde continúa la primera reunión para recabar los requerimientos.

**Participantes:** Jamie Lazar, integrante del equipo de software; Vinod Raman, miembro del equipo de software; Ed Robbins, miembro del equipo de software; Doug Miller, gerente de ingeniería de software; tres personas de mercadotecnia; un representante de ingeniería del producto, y un facilitador.

**La conversación:**

**Facilitador:** Hemos estado hablando sobre la seguridad para el acceso a la funcionalidad de CasaSegura si ha de ser posible el ingreso por internet. Me gustaría probar algo. Desarrollemos un escenario de uso para entrar a la función de seguridad.

**Jamie:** ¿Cómo?

**Facilitador:** Podríamos hacerlo de dos maneras, pero de momento mantengamos las cosas informales. Díganos (señala a una persona de mercadotecnia), ¿cómo visualiza el acceso al sistema?

**Persona de mercadotecnia:** Um... bueno, es la clase de cosa que haría si estuviera fuera de casa y tuviera que dejar entrar a alguien a ella —por ejemplo, una trabajadora doméstica o un técnico de reparaciones— que no tuviera el código de seguridad.

**Facilitador (sonríe):** Ésa es la razón por la que lo hace... dígame, ¿cómo lo haría en realidad?

**Persona de mercadotecnia:** Bueno... lo primero que necesitaría sería una PC. Entraría a un sitio web que mantendríamos para todos los usuarios de CasaSegura. Daría mi identificación de usuario y...

**Vinod (interrumpe):** La página web tendría que ser segura, encriptada, para garantizar que estuviéramos seguros y...

**Facilitador (interrumpe):** Ésa es buena información, Vinod, pero es técnica. Centrémosnos en cómo emplearía el usuario final esta capacidad, ¿está bien?

**Vinod:** No hay problema.

**Persona de mercadotecnia:** Decía que entraría a un sitio web y daría mi identificación de usuario y dos niveles de clave.

**Jamie:** ¿Qué pasa si olvido mi clave?

**Facilitador (interrumpe):** Buena observación, Jamie, pero no entraremos a ella por ahora. Lo anotaremos y la llamaremos una excepción. Estoy seguro de que habrá otras.

**Persona de mercadotecnia:** Después de que introdujera las claves, aparecería una pantalla que representaría todas las funciones de *CasaSegura*. Seleccionaría la función de seguridad del hogar. El sistema pediría que verificara quién soy, pidiendo mi dirección o número telefónico o algo así. Entonces aparecería un dibujo del panel de control del sistema de seguridad y la lista de funciones que puede realizar —activar el sistema, desactivar el sistema o desactivar uno o más sensores—. Supongo que también me permitiría reconfigurar las zonas de seguridad y otras cosas como ésa, pero no estoy seguro.

(Mientras la persona de mercadotecnia habla, Doug toma muchas notas; esto forma la base para el primer escenario informal de uso. Alternativamente, hubiera podido pedirle a la persona de mercadotecnia que escribiera el escenario, pero esto se hubiera hecho fuera de la reunión.)

#### > 5.3.4 Indagación de los productos del trabajo

Los productos del trabajo generados como consecuencia de la indagación de los requerimientos variarán en función del tamaño del sistema o producto que se va a construir. Para la mayoría de sistemas, los productos del trabajo incluyen los siguientes:



¿Qué información se produce como consecuencia de recabar los requerimientos?

- Un enunciado de la necesidad y su factibilidad.
- Un enunciado acotado del alcance del sistema o producto.
- Una lista de clientes, usuarios y otros participantes que intervienen en la indagación de los requerimientos.
- Una descripción del ambiente técnico del sistema.
- Una lista de requerimientos (de preferencia organizados por función) y las restricciones del dominio que se aplican a cada uno.
- Un conjunto de escenarios de uso que dan perspectiva al uso del sistema o producto en diferentes condiciones de operación.
- Cualesquiera prototipos desarrollados para definir requerimientos.

Cada uno de estos productos del trabajo es revisado por todas las personas que participan en la indagación de los requerimientos.

## 5.4 Desarrollo de casos de uso

En un libro que analiza cómo escribir casos de uso eficaces, Alistair Cockburn [Coc01b] afirma que “un caso de uso capta un contrato [...] [que] describe el comportamiento del sistema en distintas condiciones en las que el sistema responde a una petición de alguno de sus participantes [...].” En esencia, un caso de uso narra una historia estilizada sobre cómo interactúa un usuario final (que tiene cierto número de roles posibles) con el sistema en circunstancias específicas. La historia puede ser un texto narrativo, un lineamiento de tareas o interacciones, una descripción basada en un formato o una representación diagramática. Sin importar su forma, un caso de uso ilustra el software o sistema desde el punto de vista del usuario final.

El primer paso para escribir un caso de uso es definir un conjunto de “actores” que estarán involucrados en la historia. Los *actores* son las distintas personas (o dispositivos) que usan el sistema o producto en el contexto de la función y comportamiento que va a describirse. Los actores representan los papeles que desempeñan las personas (o dispositivos) cuando opera el sistema. Con una definición más formal, un *actor* es cualquier cosa que se comunique con el sistema o producto y que sea externo a éste. Todo actor tiene uno o más objetivos cuando utiliza el sistema.

Es importante notar que un actor y un usuario final no necesariamente son lo mismo. Un usuario normal puede tener varios papeles diferentes cuando usa el sistema, mientras que un actor representa una clase de entidades externas (gente, con frecuencia pero no siempre) que sólo tiene un papel en el contexto del caso de uso. Por ejemplo, considere al operador de una máquina (un usuario) que interactúa con la computadora de control de una celda de manufactura que contiene varios robots y máquinas de control numérico. Después de una revisión cuidadosa de los requerimientos, el software para la computadora de control requiere cuatro diferentes modos (papeles) para la interacción: modo de programación, modo de prueba, modo de vigilancia y modo de solución de problemas. Por tanto, es posible definir cuatro actores: programador, probador, vigilante y solucionador de problemas. En ciertos casos, el operador de la máquina desempeñará todos los papeles. En otros, distintas personas tendrán el papel de cada actor.

Debido a que la indagación de los requerimientos es una actividad evolutiva, no todos los actores son identificados en la primera iteración. En ésta es posible identificar a los actores principales [Jac92], y a los secundarios cuando se sabe más del sistema. Los *actores principales* interactúan para lograr la función requerida del sistema y obtienen el beneficio previsto de éste. Trabajan con el software en forma directa y con frecuencia. Los *actores secundarios* dan apoyo al sistema, de modo que los primarios puedan hacer su trabajo.

Una vez identificados los actores, es posible desarrollar casos de uso. Jacobson [Jac92] sugiere varias preguntas<sup>12</sup> que debe responder un caso de uso:

- ¿Quién es el actor principal y quién(es) el(los) secundario(s)?
- ¿Cuáles son los objetivos de los actores?
- ¿Qué precondiciones deben existir antes de comenzar la historia?
- ¿Qué tareas o funciones principales son realizadas por el actor?
- ¿Qué excepciones deben considerarse al describir la historia?
- ¿Cuáles variaciones son posibles en la interacción del actor?
- ¿Qué información del sistema adquiere, produce o cambia el actor?
- ¿Tendrá que informar el actor al sistema acerca de cambios en el ambiente externo?
- ¿Qué información desea obtener el actor del sistema?
- ¿Quiere el actor ser informado sobre cambios inesperados?

**punto clave** Los casos de uso se definen desde el punto de vista de un actor. Un actor es un papel que desempeñan las personas (usuarios) o los dispositivos cuando interactúan con el software.

**WebRef** Un artículo excelente sobre casos de uso puede descargarse desde la dirección [www.ibm.com/developerworks/webservices/library/codesign7.html](http://www.ibm.com/developerworks/webservices/library/codesign7.html)



¿Qué se necesita saber a fin de desarrollar un caso de uso eficaz?

En relación con los requerimientos básicos de *CasaSegura*, se definen cuatro actores: **propietario de la casa** (usuario), **gerente de arranque** (tal vez la misma persona que el propietario de la casa, pero en un papel diferente), **sensores** (dispositivos adjuntos al sistema) y **subsistema de vigilancia y respuesta** (estación central que vigila la función de seguridad de la casa de *CasaSegura*). Para fines de este ejemplo, consideraremos sólo al actor llamado **propietario de la casa**. Éste interactúa con la función de seguridad de la casa en varias formas distintas con el empleo del panel de control de la alarma o con una PC:

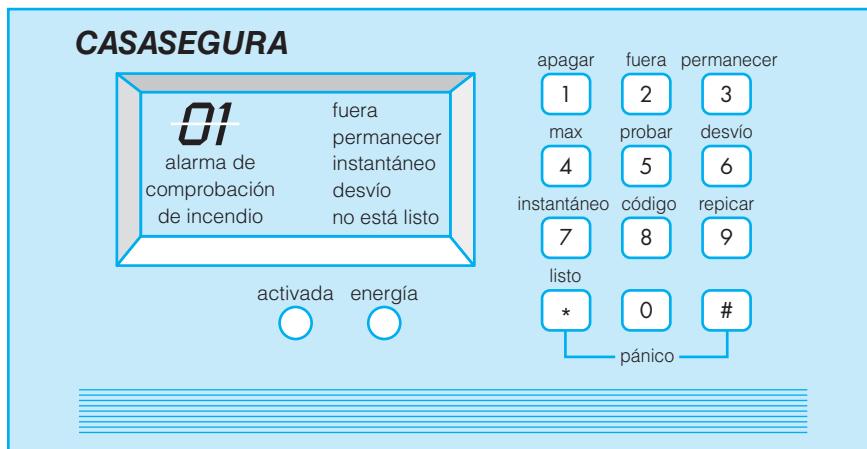
- Introduce una clave que permita todas las demás interacciones.
- Pregunta sobre el estado de una zona de seguridad.
- Interroga acerca del estado de un sensor.
- En una emergencia, oprime el botón de pánico.
- Activa o desactiva el sistema de seguridad.

Considerando la situación en la que el propietario de la casa usa el panel de control, a continuación se plantea el caso de uso básico para la activación del sistema:<sup>13</sup>

12 Las preguntas de Jacobson se han ampliado para que den una visión más completa del contenido del caso de uso.

13 Observe que este caso de uso difiere de la situación en la que se accede al sistema a través de internet. En este caso, la interacción es por medio del panel de control y no con la interfaz de usuario gráfica (GUI) que se da cuando se emplea una PC.

1. El propietario observa el panel de control de *CasaSegura* (véase la figura 5.1) para determinar si el sistema está listo para recibir una entrada. Si el sistema no está listo, se muestra el mensaje *no está listo* en la pantalla de cristal líquido y el propietario debe cerrar físicamente ventanas o puertas de modo que desaparezca dicho mensaje [el mensaje *no está listo* implica que un sensor está abierto; por ejemplo, que una puerta o ventana está abierta].
2. El propietario usa el teclado para introducir una clave de cuatro dígitos. La clave se compara con la que guarda el sistema como válida. Si la clave es incorrecta, el panel de control emitirá un sonido una vez y se reiniciará para recibir una entrada adicional. Si la clave es correcta, el panel de control espera otras acciones.
3. El propietario selecciona y teclea *permanecer* o *fuera* (véase la figura 5.1) para activar el sistema. La entrada *permanecer* activa sólo sensores perimetrales (se desactivan los sensores de detección de movimiento interior). La entrada *fuera* activa todos los sensores.
4. Cuando ocurre una activación, el propietario observa una luz roja de alarma.



**FIGURA 5.1** Panel de control de *CasaSegura*.

**consejo** Es frecuente que los casos de uso se escriban de manera informal. Sin embargo, utilice el formato que se presenta aquí para asegurar que se incluyen todos los aspectos clave.

El caso de uso básico presenta una historia de alto nivel que describe la interacción entre el actor y el sistema.

En muchas circunstancias, los casos de uso son más elaborados a fin de que brinden muchos más detalles sobre la interacción. Por ejemplo, Cockburn [Coc01b] sugiere el formato siguiente para hacer descripciones detalladas de casos de uso:

<b>Caso de uso:</b>	<i>IniciarVigilancia</i>
<b>Actor principal:</b>	Propietario.
<b>Objetivo en contexto:</b>	Preparar el sistema para que vigile los sensores cuando el propietario salga de la casa o permanezca dentro.
<b>Precondiciones:</b>	El sistema se ha programado para recibir una clave y reconocer distintos sensores.
<b>Disparador:</b>	El propietario decide “preparar” el sistema, por ejemplo, para que encienda las funciones de alarma.
<b>Escenario:</b>	<ol style="list-style-type: none"> <li>1. Propietario: observa el panel de control</li> <li>2. Propietario: introduce una clave</li> <li>3. Propietario: selecciona “permanecer” o “fuera”</li> <li>4. Propietario: observa una luz roja de alarma que indica que <i>CasaSegura</i> ha sido activada.</li> </ol>

**Excepciones:**

1. El panel de control *no está listo*: el propietario verifica todos los sensores para determinar cuáles están abiertos; los cierra.
2. La clave es incorrecta (el panel de control suena una vez): el propietario introduce la clave correcta.
3. La clave no es reconocida: debe contactarse el subsistema de vigilancia y respuesta para reprogramar la clave.
4. Se elige *permanecer*: el panel de control suena dos veces y se enciende un letrero luminoso que dice **permanecer**; se activan los sensores del perímetro.
5. Se selecciona *fuera*: el panel de control suena tres veces y se enciende un letrero luminoso que dice *fuera*; se activan todos los sensores.

**Prioridad:** Esencial, debe implementarse

**Cuándo estará disponible:** En el primer incremento

**Frecuencia de uso:** Muchas veces por día

**Canal para el actor:** A través de la interfaz del panel de control

**Actores secundarios:** Técnico de apoyo, sensores

**Canales para los actores secundarios:**

Técnico de apoyo: línea telefónica

Sensores: interfaces cableadas y frecuencia de radio

**Aspectos pendientes:**

1. ¿Debe haber una forma de activar el sistema sin usar clave o con una clave abreviada?
2. ¿El panel de control debe mostrar mensajes de texto adicionales?
3. ¿De cuánto tiempo dispone el propietario para introducir la clave a partir del momento en el que se oprime la primera tecla?
4. ¿Hay una forma de desactivar el sistema antes de que se active en realidad?

Los casos de uso para otras interacciones de **proprietario** se desarrollarían en una forma similar. Es importante revisar con cuidado cada caso de uso. Si algún elemento de la interacción es ambiguo, es probable que la revisión del caso de uso lo detecte.



### Desarrollo de un diagrama de caso de uso de alto nivel

**La escena:** Sala de juntas, continúa la reunión para recabar los requerimientos.

**Participantes:** Jamie Lazar, miembro del equipo de software; Vinod Roman, integrante del equipo de software; Ed Robbins, integrante del equipo de software; Doug Miller, gerente de ingeniería de software; tres miembros de mercadotecnia; un representante de ingeniería del producto; un facilitador.

**La conversación:**

**Facilitador:** Hemos pasado un buen tiempo hablando de la función de seguridad del hogar de CasaSegura. Durante el receso hice un diagrama de caso de uso para resumir los escenarios importantes que forman parte de esta función. Veámoslo.

(Todos los asistentes observan la figura 5.2.)

**Jamie:** Estoy aprendiendo la notación UML.<sup>14</sup> Veo que la función de seguridad del hogar está representada por el rectángulo grande con óvalos en su interior, ¿verdad? ¿Y los óvalos representan los casos de uso que hemos escrito?

**Facilitador:** Sí. Y las figuras pegadas representan a los actores —personas o cosas que interactúan con el sistema según los describe el caso de uso...—; ¡ah! usé el cuadrado para representar un actor que no es persona... en este caso, sensores.

## Doug: ¿Es válido eso en UML?

**Facilitador:** La legalidad no es lo importante. El objetivo es comunicar información. Veo que usar una figura humana para representar un equipo sería erróneo. Así que adapté las cosas un poco. No pienso que genere problemas.

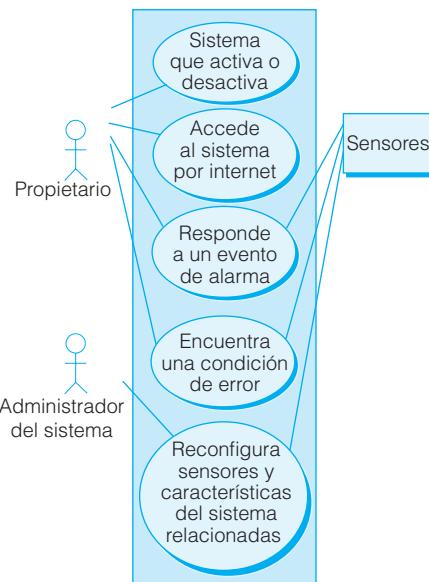
**Vinod:** Está bien, entonces tenemos narraciones de casos de uso para cada óvalo. ¿Necesitamos desarrollarlas con base en los formatos sobre los que he leído?

**Facilitador:** Es probable, pero eso puede esperar hasta que hayamos considerado otras funciones de *CasaSegura*.

**Persona de mercadotecnia:** Esperen, he estado observando este diagrama y de pronto me doy cuenta de que hemos olvidado algo.

**Facilitador:** ¿De verdad? Dime, ¿qué hemos olvidado?

(La reunión continúa.)



**FIGURA 5.2** Diagrama de caso de uso de UML para la función de seguridad del hogar de *CasaSegura*.

<sup>14</sup> En el apéndice 1 se presenta un breve método de aprendizaje de UML para aquellos lectores que no estén familiarizados con dicha notación.



## Desarrollo de un caso de uso

**Objetivo:** Ayudar a desarrollar casos de uso proporcionando formatos y mecanismos automatizados para evaluar la claridad y consistencia.

**Mecánica:** La mecánica de las herramientas varía. En general, las herramientas para casos de uso dan formatos con espacios en blanco para ser llenados y crear así casos eficaces. La mayor parte de la funcionalidad de los casos de uso está incrustada en un conjunto más amplio de funciones de ingeniería de los requerimientos.

### Herramientas representativas<sup>15</sup>

La gran mayoría de herramientas de análisis del modelado basadas en UML dan apoyo tanto de texto como gráfico para el desarrollo y modelado de casos de uso.

*Objects by Design*

([www.objectsbydesign.com/tools/umltools\\_byCompany.html](http://www.objectsbydesign.com/tools/umltools_byCompany.html)) proporciona vínculos exhaustivos con herramientas de este tipo.

Herramientas de software

## 5.5 Elaboración del modelo de los requerimientos<sup>16</sup>

El objetivo del modelo del análisis es describir los dominios de información, función y comportamiento que se requieren para un sistema basado en computadora. El modelo cambia en forma dinámica a medida que se aprende más sobre el sistema por construir, y otros participantes comprenden más lo que en realidad requieren. Por esa razón, el modelo del análisis es una fotografía de los requerimientos en cualquier momento dado. Es de esperar que cambie.

A medida que evoluciona el modelo de requerimientos, ciertos elementos se vuelven relativamente estables, lo que da un fundamento sólido para diseñar las tareas que sigan. Sin embargo, otros elementos del modelo son más volátiles, lo que indica que los participantes todavía no entienden bien los requerimientos para el sistema. En los capítulos 6 y 7 se presentan en detalle el modelo del análisis y los métodos que se usan para construirlo. En las secciones siguientes se da un panorama breve.

### > 5.5.1 Elementos del modelo de requerimientos

Hay muchas formas diferentes de concebir los requerimientos para un sistema basado en computadora. Algunos profesionales del software afirman que es mejor seleccionar un modo de representación (por ejemplo, el caso de uso) y aplicarlo hasta excluir a todos los demás. Otros piensan que es más benéfico usar cierto número de modos de representación distintos para ilustrar el modelo de requerimientos. Los modos diferentes de representación fuerzan a considerar los requerimientos desde distintos puntos de vista, enfoque que tiene una probabilidad mayor de detectar omisiones, inconsistencia y ambigüedades.

Los elementos específicos del modelo de requerimientos están determinados por el método de análisis de modelado (véanse los capítulos 6 y 7) que se use. No obstante, la mayoría de modelos tiene en común un conjunto de elementos generales.

<sup>15</sup> Las herramientas mencionadas aquí no son obligatorias, sino una muestra de las que hay en esta categoría. En la mayoría de casos, los nombres de las herramientas son marcas registradas por sus respectivos desarrolladores.

<sup>16</sup> En este libro se usan como sinónimos las expresiones *modelar el análisis* y *modelar los requerimientos*. Ambos se refieren a representaciones de los dominios de la información, funcional y de comportamiento que describen los requerimientos del problema.

**consejo** Siempre es buena idea involucrar a los participantes. Una de las mejores formas de lograrlo es hacer que cada uno escriba casos de uso que narren el modo en el que se utilizará el software.

**consejo** Una forma de aislar las clases es buscar sustantivos descriptivos en un caso de usuario expresado con texto. Al menos algunos de ellos serán candidatos cercanos. Sobre esto se habla más en el capítulo 8.

**punto clave** Un estado es un modo de comportamiento observable desde el exterior. Los estímulos externos ocasionan transiciones entre los estados.

**Elementos basados en el escenario.** El sistema se describe desde el punto de vista del usuario con el empleo de un enfoque basado en el escenario. Por ejemplo, los casos de uso básico (véase la sección 5.4) y sus diagramas correspondientes de casos de uso (véase la figura 5.2) evolucionan hacia otros más elaborados que se basan en formatos. Los elementos del modelo de requerimientos basados en el escenario con frecuencia son la primera parte del modelo en desarrollo. Como tales, sirven como entrada para la creación de otros elementos de modelado. La figura 5.3 ilustra un diagrama de actividades UML<sup>17</sup> para indagar los requerimientos y representarlos con el empleo de casos de uso. Se aprecian tres niveles de elaboración que culminan en una representación basada en el escenario.

**Elementos basados en clases.** Cada escenario de uso implica un conjunto de objetos que se manipulan cuando un actor interactúa con el sistema. Estos objetos se clasifican en clases: conjunto de objetos que tienen atributos similares y comportamientos comunes. Por ejemplo, para ilustrar la clase **Sensor** de la función de seguridad de *CasaSegura* (véase la figura 5.4), puede utilizarse un diagrama de clase UML. Observe que el diagrama enumera los atributos de los sensores (por ejemplo, nombre, tipo, etc.) y las operaciones (por ejemplo, *identificar* y *permitir*) que se aplican para modificarlos. Además de los diagramas de clase, otros elementos de modelado del análisis ilustran la manera en la que las clases colaboran una con otra y las relaciones e interacciones entre ellas. Esto se analiza con más detalle en el capítulo 7.

**Elementos de comportamiento.** El comportamiento de un sistema basado en computadora tiene un efecto profundo en el diseño que se elija y en el enfoque de implementación que se aplique. Por tanto, el modelo de requerimientos debe proveer elementos de modelado que ilustren el comportamiento.

El *diagrama de estado* es un método de representación del comportamiento de un sistema que ilustra sus estados y los eventos que ocasionan que el sistema cambie de estado. Un *estado* es cualquier modo de comportamiento observable desde el exterior. Además, el diagrama de estado indica acciones (como la activación de un proceso, por ejemplo) tomadas como consecuencia de un evento en particular.

Para ilustrar el uso de un diagrama de estado, considere el software incrustado dentro del panel de control de *CasaSegura* que es responsable de leer las entradas que hace el usuario. En la figura 5.5 se presenta un diagrama de estado UML simplificado.

Además de las representaciones de comportamiento del sistema como un todo, también es posible modelar clases individuales. Sobre esto se presentan más análisis en el capítulo 7.

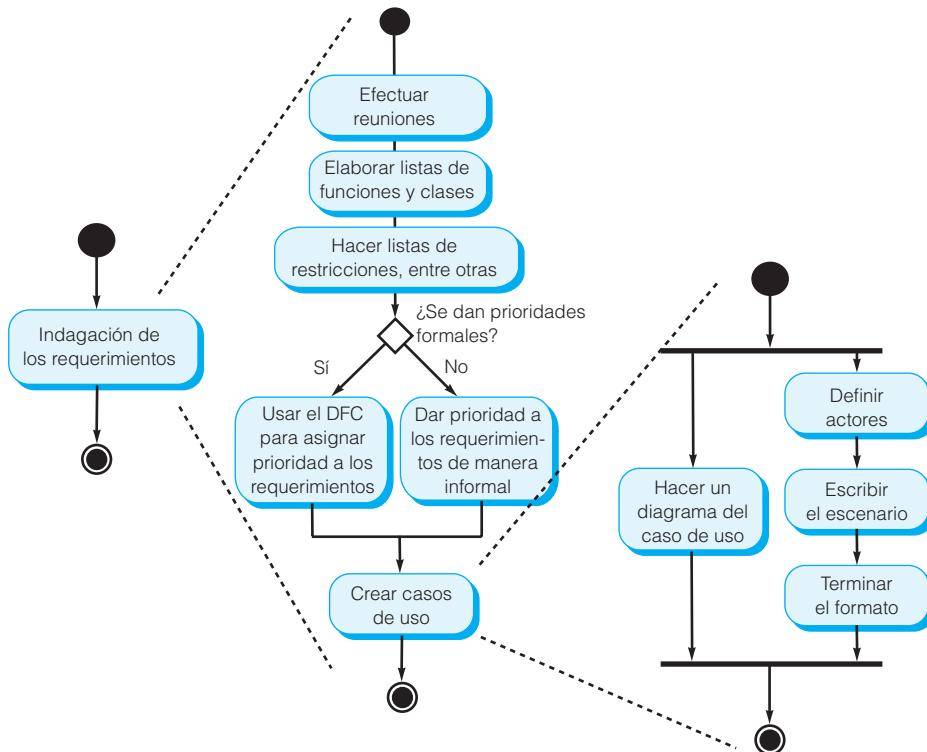
**Elementos orientados al flujo.** La información se transforma cuando fluye a través de un sistema basado en computadora. El sistema acepta entradas en varias formas, aplica funciones para transformarla y produce salidas en distintos modos. La entrada puede ser una señal de control transmitida por un transductor, una serie de números escritos con el teclado por un operador humano, un paquete de información enviado por un enlace de red o un archivo grande de datos recuperado de un almacenamiento secundario. La transformación quizás incluya una sola comparación lógica, un algoritmo numérico complicado o un enfoque de regla de inferencia para un sistema experto. La salida quizás encienda un diodo emisor de luz o genere un informe de 200 páginas. En efecto, es posible crear un modelo del flujo para cualquier sistema basado en computadora, sin importar su tamaño y complejidad. En el capítulo 7 se hace un análisis más detallado del modelado del flujo.

## 5.5.2 Patrones de análisis

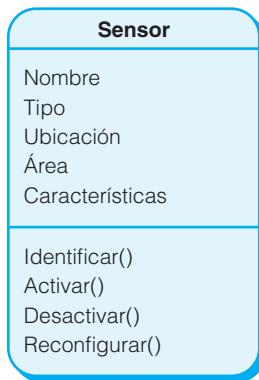
Cualquiera que haya hecho la ingeniería de los requerimientos en varios proyectos de software ha observado que ciertos problemas son recurrentes en todos ellos dentro de un dominio de aplicación específico.<sup>18</sup> Estos *patrones de análisis* [Fow97] sugieren soluciones (por ejemplo, una clase, función

<sup>17</sup> En el apéndice 1 se presenta un instructivo breve sobre UML, para aquellos lectores que no estén familiarizados con dicha notación.

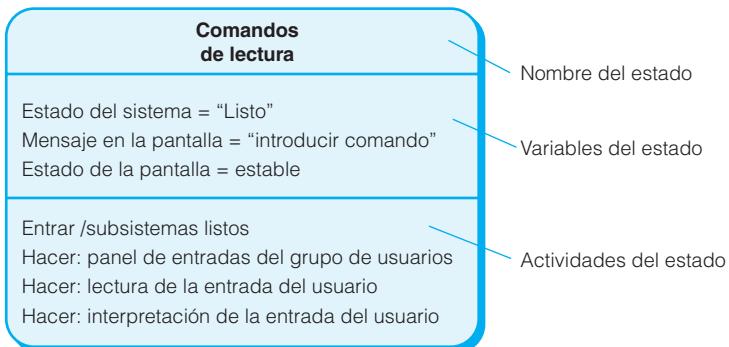
<sup>18</sup> En ciertos casos, los problemas vuelven a suceder sin importar el dominio de la aplicación. Por ejemplo, son comunes las características y funciones usadas para resolver problemas de la interfaz de usuario sin importar el dominio de la aplicación en consideración.



**FIGURA 5.3**  
Diagramas de actividad del UML para indagar los requerimientos.



**FIGURA 5.4**  
Diagrama de clase para un sensor.



**FIGURA 5.5**  
Notación UML del diagrama de estado.



## Modelado preliminar del comportamiento

**La escena:** Sala de juntas, continúa la reunión de requerimientos.

**Participantes:** Jamie Lazar, integrante del equipo de software; Vinod Raman, miembro del equipo de software; Ed Robbins, integrante del equipo de software; Doug Miller, gerente de ingeniería de software; tres trabajadores de mercadotecnia; un representante de ingeniería del producto, y un facilitador.

**La conversación:**

**Facilitador:** Estamos por terminar de hablar sobre la funcionalidad de seguridad del hogar de CasaSegura. Pero antes, quisiera que analizáramos el comportamiento de la función.

**Persona de mercadotecnia:** No entiendo lo que quiere decir con *comportamiento*.

**Ed (sonríe):** Es cuando le das un "tiempo fuera" al producto si se porta mal.

**Facilitador:** No exactamente. Permítanme explicarlo.

(El facilitador explica al equipo encargado de recabar los requerimientos y los fundamentos de modelado del comportamiento.)

**Persona de mercadotecnia:** Esto parece un poco técnico. No estoy seguro de ser de ayuda aquí.

**Facilitador:** Seguro que lo serás. ¿Qué comportamiento se observa desde el punto de vista de un usuario?

**Persona de mercadotecnia:** Mmm... bueno, el sistema estará *vigilando* los sensores. Leerá comandos del propietario. Mostrará su estado.

**Facilitador:** ¿Ves?, lo puedes hacer.

**Jamie:** También estará *interrogando* a la PC para determinar si hay alguna entrada desde ella, por ejemplo, un acceso por internet o información sobre la configuración.

**Vinod:** Sí, en realidad, *configurar* el sistema es un estado por derecho propio.

**Doug:** Muchachos, lo hacen bien. Pensemos un poco más... ¿hay alguna forma de hacer un diagrama de todo esto?

**Facilitador:** Sí la hay, pero la dejaremos para la próxima reunión.

o comportamiento) dentro del dominio de la aplicación que pueden volverse a utilizar cuando se modelen muchas aplicaciones.

Geyer-Schulz y Hahsler [Gey01] sugieren dos beneficios asociados con el uso de patrones de análisis:

En primer lugar, los patrones de análisis aceleran el desarrollo de los modelos de análisis abstracto que capturan los principales requerimientos del problema concreto, debido a que proveen modelos de análisis reutilizables con ejemplos, así como una descripción de sus ventajas y limitaciones. En segundo lugar, los patrones de análisis facilitan la transformación del modelo de análisis en un modelo del diseño, sugiriendo patrones de diseño y soluciones confiables para problemas comunes.

Los patrones de análisis se integran en el modelo del análisis, haciendo referencia al nombre del patrón. También se guardan en un medio de almacenamiento de modo que los ingenieros de requerimientos usen herramientas de búsqueda para encontrarlos y aplicarlos. La información sobre el patrón de análisis (y otros tipos de patrones) se presenta en un formato estándar [Gey01]<sup>19</sup> que se

<sup>19</sup> En la bibliografía existen varias propuestas de formatos para patrones. Si el lector tiene interés, consulte [Fow97], [Gam95], [Yac03] y [Bus07], entre muchas otras fuentes.

estudia con más detalle en el capítulo 12. En el capítulo 7 se dan ejemplos de patrones de análisis y más detalles de este tema.

## 5.6 Requerimientos de las negociaciones

En un contexto ideal de la ingeniería de los requerimientos, las tareas de concepción, indagación y elaboración determinan los requerimientos del cliente con suficiente detalle como para avanzar hacia las siguientes actividades de la ingeniería de software. Desafortunadamente, esto rara vez ocurre. En realidad, se tiene que entrar en negociaciones con uno o varios participantes. En la mayoría de los casos, se pide a éstos que evalúen la funcionalidad, desempeño y otras características del producto o sistema, en contraste con el costo y el tiempo para entrar al mercado. El objetivo de esta negociación es desarrollar un plan del proyecto que satisfaga las necesidades del participante y que al mismo tiempo refleje las restricciones del mundo real (por ejemplo, tiempo, personas, presupuesto, etc.) que se hayan establecido al equipo del software.

Las mejores negociaciones buscan un resultado “ganar-ganar”.<sup>20</sup> Es decir, los participantes ganan porque obtienen el sistema o producto que satisface la mayoría de sus necesidades y usted (como miembro del equipo de software) gana porque trabaja con presupuestos y plazos realistas y asequibles.

Boehm [Boe98] define un conjunto de actividades de negociación al principio de cada iteración del proceso de software. En lugar de una sola actividad de comunicación con el cliente, se definen las actividades siguientes:

1. Identificación de los participantes clave del sistema o subsistema.
2. Determinación de las “condiciones para ganar” de los participantes.
3. Negociación de las condiciones para ganar de los participantes a fin de reconciliarlas en un conjunto de condiciones ganar-ganar para todos los que intervienen (incluso el equipo de software).

La realización exitosa de estos pasos iniciales lleva a un resultado ganar-ganar, que se convierte en el criterio clave para avanzar hacia las siguientes actividades de la ingeniería de software.



### El arte de la negociación

Aprender a negociar con eficacia le servirá en su vida personal y técnica. Es útil considerar los lineamientos que siguen:

1. *Reconocer que no es una competencia.* Para tener éxito, ambas partes tienen que sentir que han ganado o logrado algo. Las dos tienen que llegar a un compromiso.
2. *Mapear una estrategia.* Decidir qué es lo que le gustaría lograr; qué quiere obtener la otra parte y cómo hacer para que ocurran las dos cosas.
3. *Escuchar activamente.* No trabaje en la formulación de su respuesta mientras la otra parte esté hablando. Escúchela. Es probable que obtenga conocimientos que lo ayuden a negociar mejor su posición.
4. *Centrarse en los intereses de la otra parte.* Si quiere evitar conflictos, no adopte posiciones inamovibles.
5. *No lo tome en forma personal.* Céntrese en el problema que necesita resolverse.
6. *Sea creativo.* Si están empantanados, no tenga miedo de pensar fuera de los moldes.
7. *Esté listo para comprometerse.* Una vez que se llegue a un acuerdo, no titubee; comprométase con él y címplalo.

Información

**cita:** “Un compromiso es el arte de dividir un pastel en forma tal que todos crean que tienen el trozo mayor.”

**Ludwig Erhard**

**WebRef** Un artículo breve sobre la negociación para los requerimientos de software puede descargarse desde la dirección [www.alexander-egyed.com/publications/Software\\_Requirements\\_Negotiation-Some\\_Lessons\\_Learned.html](http://www.alexander-egyed.com/publications/Software_Requirements_Negotiation-Some_Lessons_Learned.html)

<sup>20</sup> Se han escrito decenas de libros acerca de las aptitudes para negociar (por ejemplo [Lew06], [Rai06] y [Fis06]). Es una de las aptitudes más importantes que pueda aprender. Lea alguno.



## El principio de una negociación

**La escena:** Oficina de Lisa Pérez, después de la primera reunión para recabar los requerimientos.

**Participantes:** Doug Miller, gerente de ingeniería de software, y Lisa Pérez, gerente de mercadotecnia.

### La conversación:

**Lisa:** Pues escuché que la primera reunión salió realmente bien.

**Doug:** En realidad, sí. Enviste buenos representantes... contribuyeron de verdad.

**Lisa (sonríe):** Sí; en realidad me dijeron que habían entrado y que no había sido una "actividad que les despejara la cabeza".

**Doug (ríe):** La próxima vez me aseguraré de quitarme la vena tecnológica... Mira, Lisa, creo que tenemos un problema para llegar a toda esa funcionalidad del sistema de seguridad para el hogar en las fechas que propone tu dirección. Sé que aún es temprano, pero hice un poco de planeación sobre las rodillas y...

**Lisa (con el ceño fruncido):** Lo debemos tener para esa fecha, Doug. ¿De qué funcionalidad hablas?

**Doug:** Supongo que podemos tener la funcionalidad completa en la fecha establecida, pero tendríamos que retrasar el acceso por internet hasta el segundo incremento.

**Lisa:** Doug, es el acceso por internet lo que da a CasaSegura su "súper" atractivo. Toda nuestra campaña de publicidad va a girar alrededor de eso. Lo tenemos que tener...

**Doug:** Entiendo la situación, de verdad. El problema es que para dar acceso por internet tendríamos que tener un sitio web por completo seguro y en operación. Esto requiere tiempo y personal. También tenemos que elaborar mucha funcionalidad adicional en la primera entrega... no creo que podamos hacerlo con los recursos que tenemos.

**Lisa (todavía frunce el ceño):** Ya veo, pero tienes que imaginar una manera de hacerlo. Tiene importancia crítica para las funciones de seguridad del hogar y también para otras... éstas podrían esperar hasta las siguientes entregas... estoy de acuerdo con eso.

Lisa y Doug parecen estar en suspenso, pero todavía deben negociar una solución a este problema. ¿Pueden "ganar" los dos en este caso? Si usted fuera el mediador, ¿qué sugeriría?

## 5.7 Validación de los requerimientos

A medida que se crea cada elemento del modelo de requerimientos, se estudia para detectar inconsistencias, omisiones y ambigüedades. Los participantes asignan prioridades a los requerimientos representados por el modelo y se agrupan en paquetes de requerimientos que se implementarán como incrementos del software. La revisión del modelo de requerimientos aborda las preguntas siguientes:

¿Cuando se revisan los requerimientos, ¿qué preguntas deben plantearse?

- ¿Es coherente cada requerimiento con los objetivos generales del sistema o producto?
- ¿Se han especificado todos los requerimientos en el nivel apropiado de abstracción? Es decir, ¿algunos de ellos tienen un nivel de detalle técnico que resulta inapropiado en esta etapa?
- El requerimiento, ¿es realmente necesario o representa una característica agregada que tal vez no sea esencial para el objetivo del sistema?
- ¿Cada requerimiento está acotado y no es ambiguo?

- ¿Tiene atribución cada requerimiento? Es decir, ¿hay una fuente (por lo general una individual y específica) clara para cada requerimiento?
- ¿Hay requerimientos en conflicto con otros?
- ¿Cada requerimiento es asequible en el ambiente técnico que albergará el sistema o producto?
- Una vez implementado cada requerimiento, ¿puede someterse a prueba?
- El modelo de requerimientos, ¿refleja de manera apropiada la información, la función y el comportamiento del sistema que se va a construir?
- ¿Se ha “particionado” el modelo de requerimientos en forma que exponga información cada vez más detallada sobre el sistema?
- ¿Se ha usado el patrón de requerimientos para simplificar el modelo de éstos? ¿Se han validado todos los patrones de manera apropiada? ¿Son consistentes todos los patrones con los requerimientos del cliente?

Éstas y otras preguntas deben plantearse y responderse para garantizar que el modelo de requerimientos es una reflexión correcta sobre las necesidades del participante y que provee un fundamento sólido para el diseño.



## Resumen

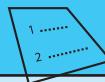
Las tareas de la ingeniería de requerimientos se realizan para establecer un fundamento sólido para el diseño y la construcción. La ingeniería de requerimientos ocurre durante las actividades de comunicación y modelado que se hayan definido para el proceso general del software. Los miembros del equipo de software llevan a cabo siete funciones de ingeniería de requerimientos: concepción, indagación, elaboración, negociación, especificación, validación y administración.

En la concepción del proyecto, los participantes establecen los requerimientos básicos del problema, definen las restricciones generales del proyecto, así como las características y funciones principales que debe presentar el sistema para cumplir sus objetivos. Esta información se mejora y amplía durante la indagación, actividad en

la que se recaban los requerimientos y que hace uso de reuniones que lo facilitan, DFC y el desarrollo de escenarios de uso.

La elaboración amplía aún más los requerimientos en un modelo: una colección de elementos basados en escenarios, clases y comportamiento, y orientados al flujo. El modelo hace referencia a patrones de análisis: soluciones para problemas de análisis que se ha observado que son recurrentes en diferentes aplicaciones.

Conforme se identifican los requerimientos y se crea su modelo, el equipo de software y otros participantes negocian la prioridad, la disponibilidad y el costo relativo de cada requerimiento. Además, se valida cada requerimiento y su modelo como un todo comparado con las necesidades del cliente a fin de garantizar que va a construirse el sistema correcto.



## Problemas y puntos por evaluar

**5.1.** ¿Por qué muchos desarrolladores de software no ponen atención suficiente a la ingeniería de requerimientos? ¿Existen algunas circunstancias que puedan ignorarse?

**5.2.** El lector tiene la responsabilidad de indagar los requerimientos de un cliente que dice estar demasiado ocupado para tener una reunión. ¿Qué debe hacer?

**5.3.** Analice algunos de los problemas que ocurren cuando los requerimientos deben indagarse para tres o cuatro clientes distintos.

**5.4.** ¿Por qué se dice que el modelo de requerimientos representa una fotografía instantánea del sistema en el tiempo?

**5.5.** Suponga que ha convencido al cliente (es usted muy buen vendedor) para que esté de acuerdo con todas las demandas que usted hace como desarrollador. ¿Eso lo convierte en un gran negociador? ¿Por qué?

**5.6.** Desarrolle al menos tres “preguntas libres de contexto” adicionales que podría plantear a un participante durante la concepción.

**5.7.** Desarrolle un “kit” para recabar requerimientos. Debe incluir un conjunto de lineamientos a fin de llevar a cabo la reunión para recabar requerimientos y los materiales que pueden emplearse para facilitar la creación de listas y otros objetos que ayuden a definir los requerimientos.

**5.8.** Su profesor formará grupos de cuatro a seis estudiantes. La mitad de ellos desempeñará el papel del departamento de mercadotecnia y la otra mitad adoptará el del equipo para la ingeniería de software. Su trabajo es definir los requerimientos para la función de seguridad de *CasaSegura* descrita en este capítulo. Efectúe una reunión para recabar los requerimientos con el uso de los lineamientos presentados en este capítulo.

**5.9.** Desarrolle un caso de uso completo para una de las actividades siguientes:

- a) Hacer un retiro de efectivo en un cajero automático.
- b) Usar su tarjeta de crédito para pagar una comida en un restaurante.
- c) Comprar acciones en la cuenta en línea de una casa de bolsa.
- d) Buscar libros (sobre un tema específico) en una librería en línea.
- e) La actividad que especifique su profesor.

**5.10.** ¿Qué representan las “excepciones” en un caso de uso?

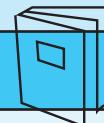
**5.11.** Describa con sus propias palabras lo que es un *patrón de análisis*.

**5.12.** Con el formato presentado en la sección 5.5.2, sugiera uno o varios patrones de análisis para los siguientes dominios de aplicación:

- a) Software de contabilidad.
- b) Software de correo electrónico.
- c) Navegadores de internet.
- d) Software de procesamiento de texto.
- e) Software para crear un sitio web.
- f) El dominio de aplicación que diga su profesor.

**5.13.** ¿Qué significa *ganar-ganar* en el contexto de una negociación durante la actividad de ingeniería de los requerimientos?

**5.14.** ¿Qué piensa que pasa cuando la validación de los requerimientos detecta un error? ¿Quién está involucrado en su corrección?



## Lecturas adicionales y fuentes de información

La ingeniería de requerimientos se estudia en muchos libros debido a su importancia crítica para la creación exitosa de cualquier sistema basado en computadoras. Hood *et al.* (*Requirements Management*, Springer, 2007) analizan varios aspectos de la ingeniería de los requerimientos que incluyen tanto la ingeniería de sistemas como la de software. Young (*The Requirements Engineering Handbook*, Artech House Publishers, 2007) presenta un análisis profundo de las tareas de la ingeniería de requerimientos. Wieggers (*More About Software Requirements*, Microsoft Press, 2006) menciona muchas técnicas prácticas para recabar y administrar los requerimientos. Hull *et al.* (*Requirements Engineering*, 2a. ed., Springer-Verlag, 2004), Bray (*An Introduction to Requirements Engineering*, Addison-Wesley, 2002), Arlow (*Requirements Engineering*, Addison-Wesley, 2001), Gilb (*Requirements Engineering*, Addison-Wesley, 2000), Graham (*Requirements Engineering and Rapid Development*, Addison-Wesley, 1999) y Sommerville y Kotonya (*Requirement Engineering: Processes and Techniques*, Wiley, 1998) son sólo algunos de los muchos libros dedicados al tema. Gottesdiener (*Requirements by Collaboration: Workshops for Defining Needs*, Addison-Wesley, 2002) proporciona una guía útil para quienes deben generar un ambiente de colaboración a fin de recabar los requerimientos con los participantes.

Lauesen (*Software Requirements: Styles and Techniques*, Addison-Wesley, 2002) presenta una recopilación exhaustiva de los métodos y notación para el análisis de requerimientos. Weiglers (*Software Requirements*, Microsoft Press, 1999) y Leffingwell *et al.* (*Managing Software Requirements: A Use Case Approach*, 2a. ed., Addison-Wesley, 2003) presentan una colección útil de las mejores prácticas respecto de los requerimientos y sugieren lineamientos prácticos para la mayoría de los aspectos del proceso de su ingeniería.

En Withall (*Software Requirement Patterns*, Microsoft Press, 2007) se describe la ingeniería de requerimientos desde un punto de vista basado en los patrones. Ploesch (*Assertions, Scenarios and Prototypes*, Springer-Verlag, 2003) analiza técnicas avanzadas para desarrollar requerimientos de software. Windle y Abreo (*Software Requirements Using the Unified Process*, Prentice-Hall, 2002) estudian la ingeniería de los requerimientos en el contexto del proceso unificado y la notación UML. Alexander y Steven (*Writing Better Requirements*, Addison-Wesley, 2002) presentan un conjunto abreviado de lineamientos para escribir requerimientos claros, representarlos como escenarios y revisar el resultado final.

Es frecuente que el modelado de un caso de uso sea el detonante para crear todos los demás aspectos del modelo de análisis. El tema lo estudian mucho Rosenberg y Stephens (*Use Case Driven Object Modeling with UML: Theory and Practice*, Apress, 2007), Denny (*Succeeding with Use Cases: Working Smart to Deliver Quality*, Addison-Wesley, 2005), Alexander y Maiden (eds.) (*Scenarios, Stories, Use Cases: Through the Systems Development Life-Cycle*, Wiley, 2004), Leffingwell *et al.* (*Managing Software Requirements: A Use Case Approach*, 2a. ed., Addison-Wesley, 2003) presentan una colección útil de las mejores prácticas sobre los requerimientos. Bittner y Spence (*Use Case Modeling*, Addison-Wesley, 2002), Cockburn [Coc01], Armour y Miller (*Advanced Use Cases Modeling: Software Systems*, Addison-Wesley, 2000) y Kulak *et al.* (*Use Cases: Requirements in Context*, Addison-Wesley, 2000) estudian la obtención de requerimientos con énfasis en el modelado del caso de uso.

En internet hay una variedad amplia de fuentes de información acerca de la ingeniería y análisis de los requerimientos. En el sitio web del libro, [www.mhhe.com/engcs/compsi/pressman/professional/olc.ser.htm](http://www.mhhe.com/engcs/compsi/pressman/professional/olc.ser.htm), se halla una lista actualizada de referencias en web que son relevantes para la ingeniería y análisis de los requerimientos.

# Tema 4

Análisis de Sistema



# 4 Análisis de sistemas

## Panorámica y objetivos del capítulo

En este capítulo, usted aprenderá más acerca de las fases de análisis de sistemas en un proyecto de desarrollo de sistemas; a saber, definición de alcance, análisis de problema, análisis de requerimientos y fases de análisis de decisión. Las primeras tres fases son llamadas en forma colectiva *análisis de sistemas*. La última fase proporciona una transición entre el análisis de sistemas y el diseño de sistemas. Usted sabrá que comprende el proceso del análisis de sistemas cuando pueda:

- Definir análisis de sistemas y relacionar el término con la definición de alcance, análisis de problema, análisis de requerimientos, diseño lógico y fases de análisis de decisión de la metodología de desarrollo de sistemas de este texto.
- Describir diversos métodos de análisis de sistemas para resolver problemas de sistemas de negocios.
- Describir la definición de alcance, análisis de problemas, análisis de requerimientos, diseño lógico y las fases de análisis de decisión en términos de los bloques de construcción de su sistema de información.
- Describir la definición de alcance, análisis de problemas, análisis de requerimientos, diseño lógico y las fases de análisis de decisión en términos de propósito, participantes, entradas, salidas, técnicas y pasos.
- Identificar en este libro los capítulos que pueden ayudarle a aprender herramientas y técnicas específicas de análisis de sistemas.

NOTA: Aunque en este capítulo se revisan algunas herramientas y técnicas del análisis de sistemas, *no* es el propósito de este capítulo enseñarlas; en él se revisa sólo el *proceso* del análisis de sistemas. Las herramientas y técnicas se enseñarán en los cinco capítulos posteriores.

## Introducción

Bob Martínez recuerda haber aprendido en la universidad que el análisis de sistemas define lo que un sistema de información necesita hacer mientras que un diseño de sistemas define la forma en que tiene que hacerlo. En ese tiempo, sonaba como un proceso simple de dos pasos. Ahora, mientras comienza a trabajar en el proyecto de sistemas de servicios de membresía de SoundStage, puede ver que hay múltiples fases y varios pasos dentro del análisis de sistemas y del diseño de sistemas.

El proyecto de SoundStage está al inicio del análisis de sistemas, en lo que Sandra, su jefa, llama la fase de definición de alcance. Después de eso, harán un análisis del problema, análisis de requerimientos y análisis de decisiones. Suena como mucho trabajo sólo para entender *qué* debe hacer el sistema. Pero este es un sistema complicado. Como dice Sandra, ¿construiría usted una casa sin un buen juego de planos?

## ¿Qué es un análisis de sistemas?

**análisis de sistemas** Técnica de solución de problemas que descompone el sistema en sus componentes para estudiar el grado en que éstos funcionan e interactúan para lograr su propósito.

**diseño de sistemas** Técnica complementaria (de la de análisis de sistemas) de solución de problemas que reensambla los componentes de un sistema en el sistema completo, con la esperanza de mejorarlo. Ello puede abarcar la adición, la eliminación y el cambio de componentes en relación con el sistema original.

**análisis de sistemas de información** Las fases de desarrollo de un proyecto de desarrollo de sistemas de información que se centran principalmente en los problemas y requerimientos de negocios, con independencia de la tecnología que pueda usarse o se use para implantar una solución al problema.

**repositorio** Base de datos o directorio de archivos donde los desarrolladores de sistemas guardan toda la documentación, conocimientos y herramientas de uno o más proyectos o sistemas de información. Los depósitos usualmente se automatizan para facilitar el almacenamiento y la recuperación, así como para compartir la información.

En el capítulo 3 usted aprendió acerca del proceso de desarrollo de sistemas y, a propósito, limitamos nuestro análisis a considerar de manera breve cada fase. En este capítulo, damos un vistazo más cercano a esas fases que en forma colectiva se llaman **análisis de sistemas**. Con la definición formal al margen, el análisis de sistemas es el estudio de un sistema y sus componentes. Es un requerimiento previo para el **diseño de sistemas**, la especificación de un sistema nuevo y mejorado. Este capítulo se enfocará en el análisis de sistemas. En el capítulo 10 se hará lo mismo para el diseño de sistemas.

Al pasar de esta definición clásica del análisis de sistemas hacia algo un poco más contemporáneo, vemos que el *análisis de sistemas* es un término que en forma colectiva describe las fases iniciales del desarrollo de sistemas. En la figura 4.1 se marcaron las etapas 1 a 5 para identificar las fases de análisis de sistemas en el contexto de la ruta clásica completa para nuestra metodología *FAST* (del capítulo 3). Nunca ha habido una definición universalmente aceptada del análisis de sistemas. De hecho, nunca ha habido un acuerdo universal acerca de cuándo termina un análisis de sistema de información y cuándo comienza el diseño de sistemas de información. Para el propósito de este libro, con el **análisis de sistemas de información** se enfatizan los temas de *negocios*, no los asuntos técnicos o de implementación.

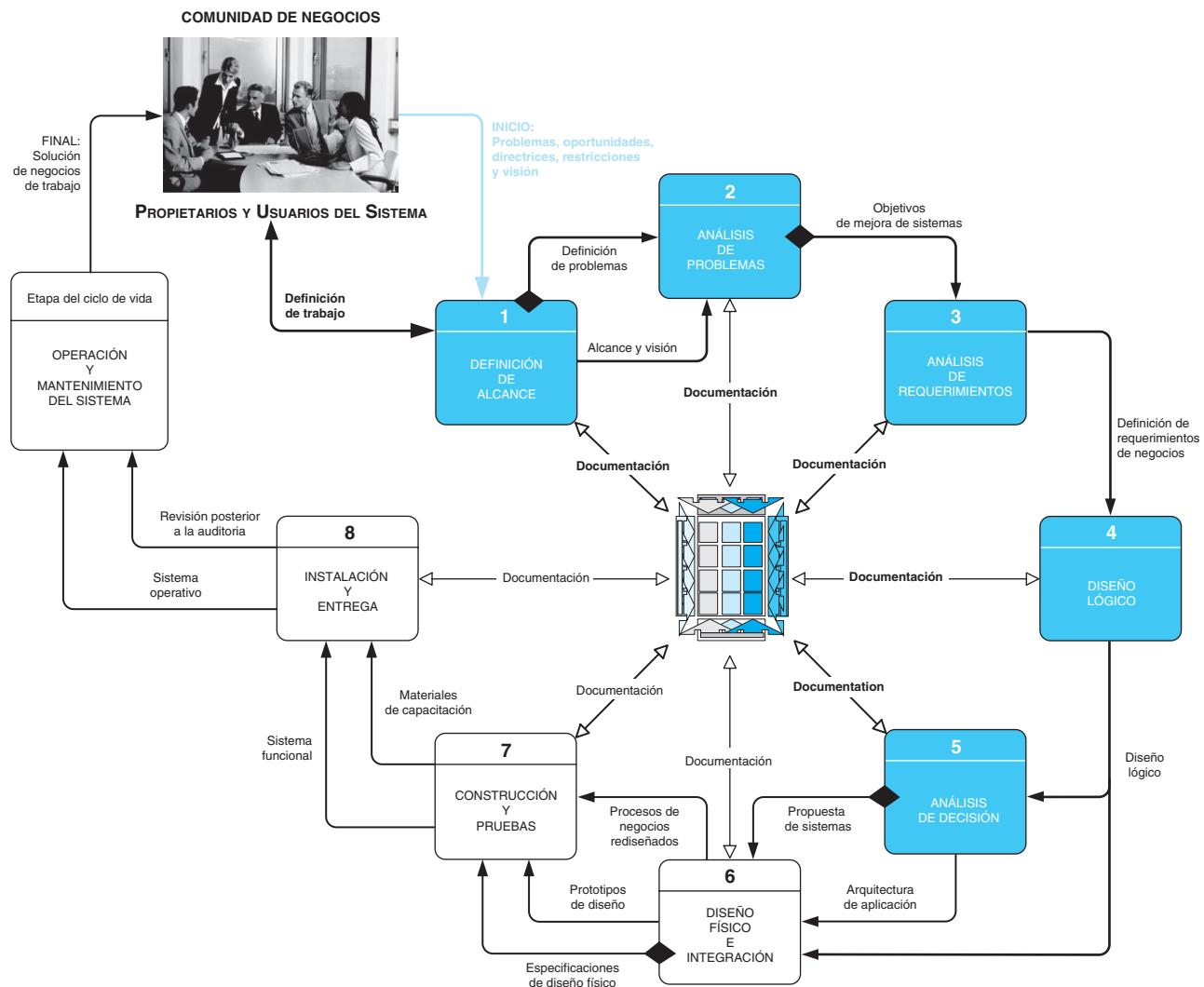
El análisis de sistemas es impulsado por los asuntos de negocios de los PROPIETARIOS DE SISTEMAS y los USUARIOS DE SISTEMAS. Por tanto, aborda los bloques de construcción de CONOCIMIENTO, PROCESO y COMUNICACIONES desde las perspectivas de los PROPIETARIOS DEL SISTEMA y los USUARIOS DE SISTEMAS. Los ANALISTAS DE SISTEMAS sirven como facilitadores del análisis de sistemas. Este contexto se ilustra en la página de inicio del capítulo que precedió a los objetivos del mismo.

La documentación y los productos obtenidos por las tareas de análisis de sistemas por lo general se almacenan en un repositorio. Éste puede ser creado para un solo proyecto o ser compartido por todos los proyectos y sistemas. Por lo general un repositorio se implementa como una combinación de lo siguiente:

- Un directorio de red de procesadores de palabras, hojas de cálculo y otros archivos generados en computadoras que contienen correspondencia de proyectos, informes y datos.
- Uno o más diccionarios o enciclopedias de herramientas CASE (como se analizó en el capítulo 3).
- Documentación impresa (como la almacenada en carpetas y bibliotecas de sistemas).
- Una interfaz de sitio Web de *intranet* para los componentes anteriores (útil para la comunicación).

De aquí en adelante, nos referiremos a los componentes en forma colectiva como **repositorio**.

En este capítulo analizaremos cada una de nuestras cinco fases de análisis de sistemas con mayor detalle. Pero primero, analicemos algunas estrategias generales para el análisis de sistemas.



**FIGURA 4.1** El contexto del análisis de sistemas

## Enfoques de análisis de sistemas

De manera fundamental, el análisis de sistemas trata acerca de la *solución de problemas*. Hay muchos métodos para ello; por lo tanto, no le debe sorprender que haya muchos enfoques para el análisis de sistemas. Estos enfoques con frecuencia son vistos como alternativas en *competencia*. En realidad, ciertas combinaciones pueden y deben complementarse entre ellas. Esto se presentó en el capítulo 3 como *métodos acelerados*. Analicemos de manera breve los diversos enfoques.

NOTA: La intención aquí es sólo desarrollar una comprensión de alto nivel. En los capítulos posteriores a esta unidad, se enseñarán las técnicas subyacentes.

### > Enfoques de análisis basados en modelos

El análisis estructurado, la ingeniería de información y el análisis orientado a objetos son ejemplos de un **análisis basado en modelos**. Este análisis utiliza imágenes para comu-

**análisis basado en modelos** Una estrategia de solución de problemas que hace énfasis en trazar modelos de sistemas de imágenes para documentar y validar los sistemas existentes o propuestos. En última instancia, el modelo de sistema se convierte en el plano para diseñar y construir un sistema mejorado.

**modelo** Representación de la realidad. Puesto que “una imagen vale más que mil palabras”, en muchos modelos se usan imágenes para representar la realidad.

**análisis estructurado** Técnica centrada en PROCESOS y operada por modelos que se usa para analizar un sistema existente, para definir los requerimientos de negocios de un nuevo sistema o para ambos objetivos. Los modelos son imágenes que ilustran los componentes del sistema: procesos, entradas, salidas y archivos.

nicar problemas de negocios, requerimientos y soluciones. Ejemplos de **modelos** con los que puede ya estar familiarizado incluyen diagramas de flujo, cuadros de estructura o jerarquías y organigramas.

En la actualidad, los enfoques basados en modelos casi siempre se resaltan por el uso de herramientas automatizadas. Algunos analistas dibujan modelos de sistemas con software gráfico de propósitos generales como *Visio* de Microsoft. Otros analistas y organizaciones requieren el uso de herramientas basadas en repositorios CASE o de elaboración de modelos como *System Architect*, *Visible Architect*, *Visible Analyst* o *Rational ROSE*. Las herramientas CASE ofrecen la ventaja del análisis consistente y completo así como una revisión de errores basada en reglas.

Los enfoques de análisis basados en modelos se presentan en las metodologías y rutas basadas en modelos que se presentaron en el capítulo 3. Analicemos de manera breve los tres enfoques más populares de análisis basados en modelos.

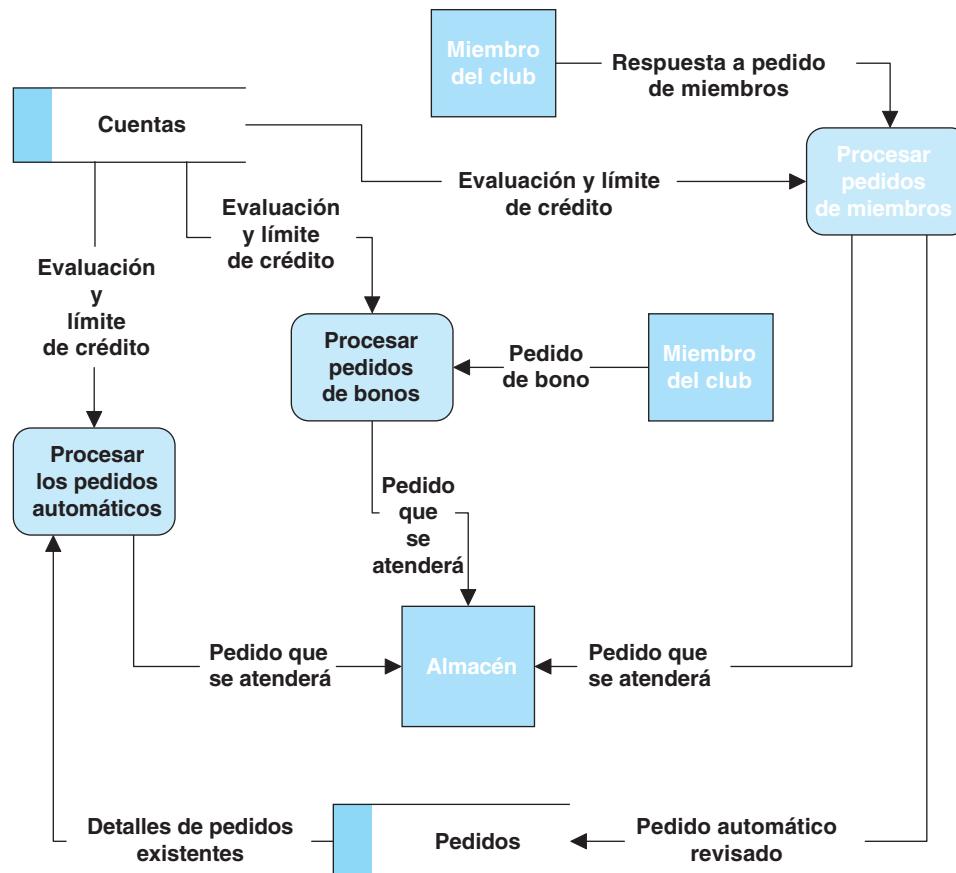
**Métodos tradicionales** Al inicio de la década de los setenta se desarrollaron diversos métodos tradicionales para el análisis y el diseño de sistemas. Uno de los primeros métodos formales, que aún es ampliamente utilizado en la actualidad es el *análisis estructurado*. El **análisis estructurado** se enfoca en el flujo de datos a través de los procesos de negocios y de software. Se dice que está *centrado en el proceso*. Por centrado en el proceso queremos decir que el énfasis está en los componentes (bloques de construcción) del PROCESO en su marco de referencia del sistema de información.

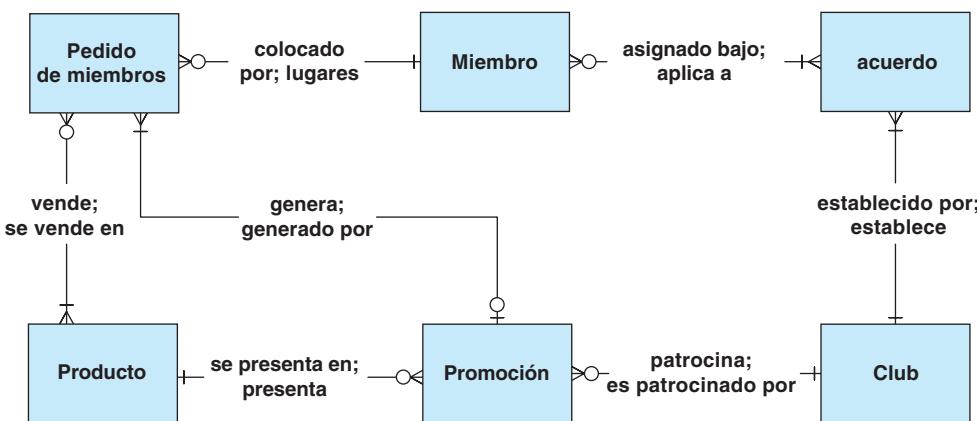
Una de las herramientas clave utilizadas para elaborar modelos de procesos es el *diagrama de flujo de datos* (figura 4.2) que describe los procesos existentes o propuestos en un sistema junto con sus entradas, salidas y datos. Los modelos muestran el flujo de datos entre y a través de los procesos y los lugares donde se almacenan los datos. Por último, estos modelos de procesos sirven como planos para que los procesos de negocios sean implementados y que el software se adquiera o se construya.

La práctica de análisis estructurado para el diseño de software ha disminuido mucho en favor de los métodos orientados a objetos. Sin embargo, la elaboración de modelos

**FIGURA 4.2**

Modelo de proceso simple (también llamado diagrama de flujo de datos)



**FIGURA 4.3**

Modelo de datos simples (también llamado diagrama de entidad-relación)

de proceso disfruta un cierto reavivamiento gracias el énfasis renovado en el rediseño de procesos de negocios, que se analizará posteriormente en este capítulo.

Otro método tradicional, llamado **ingeniería de información (IE)**, se enfoca en la estructura de datos almacenados en un sistema más que en los procesos. Por ello, se dijo que era *centrado en los datos*, al enfatizar el análisis de los requerimientos de CONOCIMIENTO (o datos). La herramienta fundamental para modelar requerimientos de datos es el diagrama de relación de entidad (figura 4.3). Los diagramas de relación de entidades aún son muy utilizados en el diseño de bases de datos de relaciones.

En un principio, la ingeniería de información era vista como un método en competencia con el análisis estructurado. Con el paso del tiempo muchas personas los hicieron complementarios; utilizan diagramas de flujo de datos para modelar los procesos de sistemas y los diagramas de relaciones de entidades para modelar un sistema de datos.

**Estrategia orientada a objetos** Los métodos tradicionales separaban las preocupaciones en forma deliberada del CONOCIMIENTO (datos) de los de PROCESOS. Aunque la mayoría de los métodos de análisis de sistemas intentaron sincronizar modelos de datos y procesos, ese intento no siempre funcionaba bien en la práctica. Las tecnologías de **objeto** han surgido desde entonces para eliminar esta separación artificial de datos y procesos. La **estrategia orientada a objetos** ve los sistemas de información no como datos y procesos, sino como una colección de objetos que encapsulan datos y procesos. Los objetos pueden contener atributos de datos. Sin embargo, la única forma de crear, leer, actualizar o eliminar los datos de un objeto es a través de uno de sus procesos incrustados (llamado métodos). Los lenguajes de programación orientados a objetos como *Java*, *C++* y los lenguajes .NET, se vuelven cada vez más populares.

La estrategia orientada a objetos tiene un conjunto completo de herramientas de elaboración de modelos conocido como Unified Modeling Language (UML). Uno de los diagramas UML de clase de objetos, se muestra en la figura 4.4. Algunas de las herramientas UML han ganado aceptación por los proyectos de sistemas incluso cuando el sistema de información no sea implementado con tecnologías orientadas a objetos.

### > Enfoques de análisis de sistemas acelerados

El desarrollo de la elaboración de prototipos de identificación y de arquitectura rápida son ejemplos de enfoques de sistemas acelerados que enfatizan la construcción de prototipos para identificar con más rapidez los requerimientos de negocios y de usuarios para un sistema nuevo. La mayoría de dichos métodos se derivan de alguna variación en la construcción de **prototipos**, muestras funcionales pero incompletas de un sistema deseado. Los prototipos sirven a la forma de pensar de “sabré lo que quiero cuando lo vea”, que es característico de muchos usuarios y administradores. Por “incompleto” queremos decir que un prototipo no incluirá la revisión de errores, validación de entrada de datos, seguridad y totalidad de proceso de una aplicación terminada. Ni tampoco estará tan pulida ni ofrecerá ayuda al usuario como en el sistema final. Pero como puede ser desarrollado con rapidez, de igual forma puede identificar los requerimientos más cruciales de nivel de negocios. A veces, los prototipos pueden evolucionar para convertirse en los sistemas finales, es decir, las aplicaciones completas.

**ingeniería de información (IE)** Una técnica operada por modelos y centrada en DATOS, pero sensible a PROCESOS, para la planeación, el análisis y el diseño de sistemas de información. Los modelos de IE son imágenes que ilustran y sincronizan los datos y procesos del sistema.

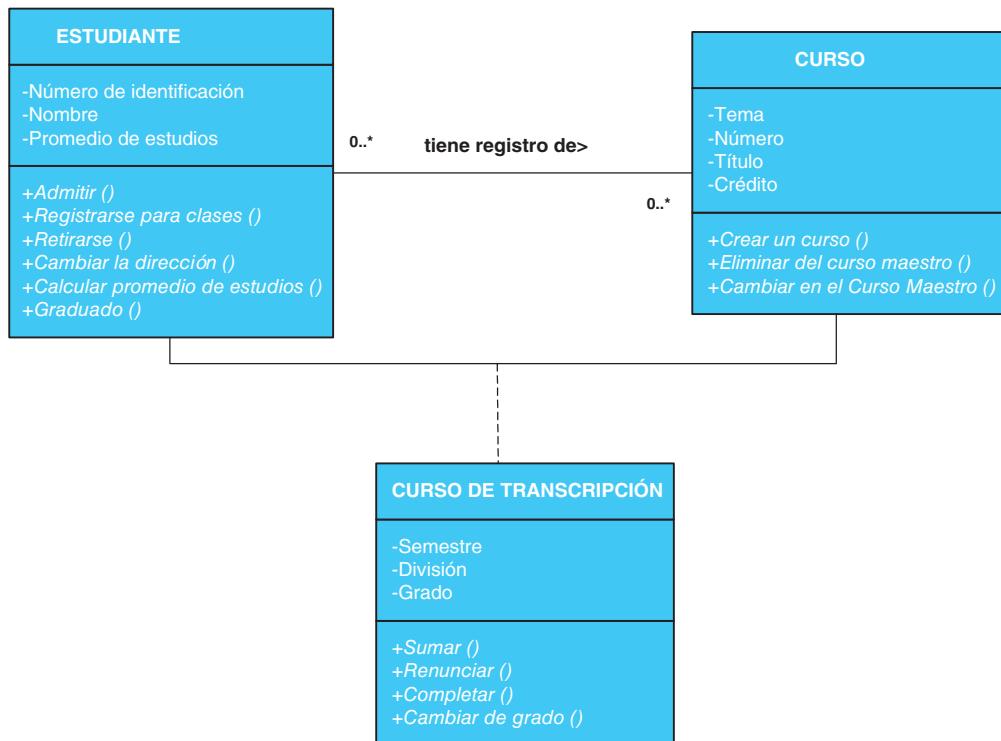
**objeto** Encapsulación de datos (llamados *propiedades*) que describen a una persona, objeto, sitio o evento, con todos los procesos (llamados *métodos*) permitidos para usar o actualizar los datos y propiedades. La única forma de tener acceso a los datos del objeto o actualizarlos es usar los procesos predefinidos del objeto.

**estrategia orientada a objetos** Técnica basada en modelos que integra los datos y procesos en conceptos llamados *objetos*. Los modelos de objetos son imágenes que ilustran los objetos del sistema desde diversas perspectivas, como la estructura, el comportamiento y las interacciones entre ellos.

**prototipo** Muestra a pequeña escala, un ejemplo incompleto pero funcional de un sistema deseado.

**FIGURA 4.4**

Modelo de objeto  
(con el estándar  
del lenguaje de  
elaboración de  
modelos unificado)



En un sentido, los métodos de análisis acelerado ponen mucho énfasis en los componentes de COMUNICACIONES en su marco de referencia de sistema de información al construir formas e informes de muestras. Al mismo tiempo, las herramientas de software utilizadas para construir prototipos también incluyen los componentes de DATOS y PROCESOS.

Estos métodos acelerados son comunes en las metodologías de desarrollo rápido de aplicaciones (rapid application development, RAD) y las rutas que fueron presentadas en el capítulo 3. Los métodos RAD requieren herramientas automatizadas mientras que algunas herramientas CASE basadas en repositorios incluyen instalaciones RAD muy simples, la mayoría de los analistas utilizan verdaderos ambientes de programación RAD, como *Powerbuilder* de Sybase, *Access* de Microsoft, *Visual Basic .NET* de Microsoft o *Websphere Studio for Application Development* de IBM (basado en *Java*).

Analicemos de manera breve dos métodos populares de análisis acelerado.

**elaboración de prototipos de identificación** Técnica usada para identificar los requerimientos de negocios de los usuarios al hacerlos reaccionar a una implantación rápida y no acabada de esos requerimientos.

**Elaboración de prototipos de identificación** La elaboración de prototipos de identificación utiliza tecnología de desarrollo rápido para ayudar a los usuarios a identificar sus requerimientos de negocios. Por ejemplo, es muy común que los analistas de sistemas utilicen una herramienta de desarrollo simple como *Access* de Microsoft para crear en forma rápida una base de datos simple, formatos de entrada de usuario e informes de muestra para solicitar respuestas de usuarios en cuanto a si la base de datos, formatos e informes representan en realidad los requerimientos de negocios. La intención es desarrollar de manera normal el nuevo sistema final en una herramienta y un lenguaje de desarrollo de aplicación más sofisticado, pero la herramienta más simple permite al analista desarrollar los prototipos de acuerdo con los requerimientos de los usuarios con una mayor rapidez.

En la elaboración de prototipos para identificación, tratamos de desalentar a los usuarios de preocuparse por la “visión y sensación” final de los prototipos de sistemas, ¡que pueden cambiarse durante el diseño del sistema! Aquí reside la crítica primaria de la elaboración de prototipos; las plantillas de software existen en las herramientas de elaboración de prototipos para generar en forma rápida algunos prototipos muy elegantes y visualmente atractivos. La desventaja es que esto puede alejar un enfoque y compromiso prematuro con el diseño representado en el prototipo. Los usuarios también pueden ser llevados a creer en forma equivocada: 1) que el sistema completo puede construirse con la misma rapidez, o 2) que herramientas como *Access* pueden utilizarse para construir el sistema final. Aunque herramientas como *Access* pueden en realidad acelerar el desarrollo de sistemas, su uso en la elaboración de prototipos es rápido sólo porque se omite la

mayor parte del detalle de la base de datos y la programación de aplicaciones requerida para una aplicación completa y segura. También, por lo general, herramientas como *Access* no pueden soportar los tamaños, número de usuarios y tráfico de red de las bases de datos que se requieren en la mayoría de las aplicaciones empresariales.

No obstante, la elaboración de prototipos de identificación es un método preferido y recomendado. Desafortunadamente, algunos analistas y desarrolladores de sistemas utilizan la elaboración de prototipos de identificación para reemplazar en forma total el diseño basado en modelos, sólo para darse cuenta lo que los verdaderos ingenieros han sabido durante años: No se puede elaborar un prototipo sin cierta cantidad de un diseño más formal... y es aquí donde entra el análisis rápido con arquitectura.

**Análisis rápido de arquitectura** El **análisis rápido de arquitectura** es un método de análisis acelerado que también construye modelos de sistemas. El análisis rápido de arquitectura se hace posible mediante la tecnología de **ingeniería inversa** que se incluye en muchas herramientas automatizadas como CASE y lenguajes de programación (como se presentaron en el capítulo 3). Las herramientas de ingeniería inversa generan modelos de sistemas de las aplicaciones de software existentes o los prototipos de sistemas. Los modelos de sistemas resultantes pueden entonces ser editados y mejorados por los analistas de sistemas y los usuarios para proporcionar un plan para un sistema nuevo y mejorado. Debe ser evidente que el análisis rápido con arquitectura es una mezcla de los enfoques basados en modelos y de análisis acelerado.

Hay dos técnicas diferentes para aplicar un análisis de arquitectura rápido:

- La mayoría de los sistemas ya han sido automatizados hasta cierto grado y existen como sistema de información de heredado. Muchas herramientas CASE pueden leer las estructuras de base de datos subyacentes y/o los programas de aplicación, y luego, con ingeniería inversa, regresarlos en diferentes modelos de sistemas. Esos modelos sirven como punto de partida para definir un análisis de requerimientos de usuario basado en modelos.
- Si los prototipos han sido construidos en herramientas como *Access* o *Visual Basic* de Microsoft, esos prototipos a veces pueden ser revertidos con la ingeniería inversa a su equivalente en modelos de sistemas. Estos últimos por lo general se prestan para analizar los requerimientos de los usuarios en cuanto a su consistencia, totalidad, estabilidad, escalabilidad y flexibilidad al cambio futuro. También, los modelos de sistemas con frecuencia pueden ser cambiados por la ingeniería hacia adelante por medio de las mismas herramientas CASE y los ADE (ambientes de desarrollo de aplicación) en las bases de datos y plantillas de aplicación o los esqueletos que utilizarán las robustas bases de datos de nivel empresarial y tecnología de programación.

Ambas técnicas abordan el tema anterior de que los ingenieros rara vez elaboran prototipos en la ausencia total de un diseño más formal y al mismo tiempo, preservan las ventajas de acelerar las fases de análisis de sistemas.

## > Métodos para identificación de requerimientos

Los métodos de análisis de sistemas acelerados y el basado en modelos intentan expresar los requerimientos de usuarios para un nuevo sistema, ya sea como modelos o como prototipos. Pero ambos métodos son, a su vez, dependientes de la necesidad más sutil de identificar y administrar en realidad esos requerimientos. Más aún, los requerimientos para los sistemas dependen de la capacidad de los analistas para identificar los problemas y oportunidades que existen en el sistema actual, por tanto, los analistas deben volverse hábiles para identificar problemas, oportunidades y requerimientos. En consecuencia, todas las estrategias para análisis de sistemas requieren alguna forma de **identificación de requerimientos**. Revisemos de manera breve un par de estrategias para identificación de requerimientos.

**Técnicas de identificación de hechos** La **identificación de hechos** es una habilidad esencial para todos los analistas de sistemas. Las técnicas de identificación de hechos que se abordan en este texto (de hecho, en el siguiente capítulo) incluyen:

- El muestreo de la documentación existente, informes, formatos, archivos, bases de datos y memorandos.
- Investigación de bibliografía relevante, sondeo en el mercado de otras soluciones y visitas a sitios.

**análisis rápido de arquitectura** Estrategia que intenta derivar modelos de sistemas (como se describe con antelación en la misma sección del capítulo) a partir de sistemas existentes o prototipos de identificación.

**ingeniería inversa** Uso de tecnología que lee el código de programas a partir de bases de datos, programas de aplicación o interfaces de usuarios existentes y genera automáticamente el modelo de sistemas equivalente.

**identificación de requerimientos** Proceso que usan los analistas de sistemas para identificar o extraer problemas de sistemas y requerimientos de solución de la comunidad de usuarios.

**identificación de hechos** Proceso de recopilar información acerca de problemas, oportunidades, requerimientos de solución y prioridades del sistema. También denominado *recopilación de información*.

- Observación del sistema actual en acción y el ambiente de trabajo.
- Cuestionarios y encuestas de la administración y la comunidad de usuarios.
- Entrevistas de administradores, usuarios y personal técnico apropiado.

**Planeación conjunta de requerimientos** Las técnicas de identificación de hechos listados con anterioridad son invaluables; sin embargo, pueden consumir mucho tiempo en sus formas clásicas. De manera alternativa, la identificación de requerimientos y la administración pueden ser acelerados en forma significativa al usar una técnica de **planeación conjunta de requerimientos (joint requirements planning, JRP)**. Un analista capacitado o certificado en JRP en general desempeña el papel de *facilitador* en un taller que normalmente toma de tres a cinco días completos de trabajo. Este taller puede reemplazar semanas o meses de las clásicas juntas de identificación de hechos y su seguimiento.

Una JRP proporciona un ambiente de trabajo en el cual se aceleran todas las tareas y los productos del análisis de sistemas. Promueve una participación mejorada del PROPIETARIO DE SISTEMAS y del USUARIO DE SISTEMAS en el análisis del sistema. Pero también requiere que un facilitador con excelentes habilidades de negociación y conciliación se asegure de que todas las partes reciban las oportunidades apropiadas para contribuir al desarrollo del sistema.

La JRP por lo general se utiliza en conjunto con los métodos de análisis basados en sistemas que describimos con anterioridad y casi siempre se incorpora en las metodologías y rutas de desarrollo rápido de aplicación (RAD), que se presentaron en el capítulo 3.

### > Métodos de rediseño de procesos de negocios

Una de las aplicaciones contemporáneas más interesantes de los métodos de análisis de sistemas es el **rediseño de procesos de negocios (business process redesign, BPR)**. El interés en el BPR fue dirigido por el descubrimiento de que los sistemas de información y aplicaciones más actuales sólo han automatizado los procesos de negocios existentes e inefficientes. La burocracia automatizada sigue siendo burocracia; la automatización no necesariamente aporta valor al negocio y en verdad puede sustraer valor al negocio. Presentado en el capítulo 1, el BPR es uno de muchos tipos de proyectos disparados por las tendencias que llamamos de *administración de calidad total (total quality management, TQM)* y *mejora continua de procesos (continuos process improvement, CPI)*.

Algunos proyectos de BPR se enfocan en todos los procesos de negocios, sin importar su automatización. Cada proceso de negocios se estudia y analiza con profundidad en busca de cuellos de botella, valor de retorno y oportunidades de eliminación o agilización. Los modelos de procesos, tales como diagramas de flujo de datos (analizados con anterioridad) ayudan a las organizaciones a visualizar sus procesos. Una vez que los procesos de negocios han sido rediseñados, la mayoría de los BPR concluyen al examinar cómo sería mejor aplicada la tecnología de la información a los procesos de negocios mejorados. Esto puede crear nuevos proyectos de desarrollo de sistemas de información y aplicación para implementar o soportar los nuevos procesos de negocios.

BPR también se aplica dentro del contexto de los proyectos de desarrollo de información. Es muy común que los proyectos de IS incluyan un estudio de los procesos de negocios existentes para identificar problemas, burocracia e inefficiencias que puedan ser abordados en requerimientos para nuevos y mejorados sistemas de información y aplicaciones de cómputo.

BPR también se ha vuelto común en los proyectos de IS que estarán basados en la compra e integración de software listo para usarse (commercial off-the-shelf, COTS). La adquisición de software COTS en general requiere que un negocio adapte sus procesos de negocios para encajar en el software. Un análisis de procesos de negocios existentes durante el análisis de sistemas casi siempre es parte de dichos proyectos.

### > Estrategias de análisis de sistemas FAST

Al igual que la mayoría de las metodologías comerciales, nuestra metodología hipotética **FAST** no impone un solo método en los analistas de sistemas. En su lugar, integra todos los métodos populares presentados en los párrafos anteriores en una colección de **métodos acelerados**. El caso de estudio de SoundStage demostrará estos métodos en el contexto de una primera asignación de un analista de sistemas. Las técnicas de análisis de sistemas serán aplicadas dentro del marco de referencia de:

#### planeación conjunta de requerimientos (JRP)

Uso de talleres facilitados para reunir a todos los propietarios, usuarios y analistas de un sistema y a ciertos diseñadores y constructores de sistemas con el fin de realizar conjuntamente el análisis de sistemas. La JRP por lo general se considera como parte de un método más amplio, llamado *desarrollo conjunto de aplicaciones (JAD)*, que es una aplicación más completa de las técnicas de JRP al proceso de desarrollo de sistemas en su totalidad.

#### rediseño de procesos de negocios (BPR)

Aplicación de métodos de análisis de sistemas con el objetivo de cambiar y mejorar significativamente los procesos de negocios fundamentales de una organización, con independencia de la tecnología de la información.

#### método acelerado

Integración de diversos enfoques del análisis y diseño de sistemas para su aplicación según se considere apropiado al problema que se intenta resolver y el sistema que se está desarrollando.

- Los componentes de su sistema de información (del capítulo 2).
- Las fases de *FAST* (del capítulo 3).
- Las tareas de *FAST* que implementan una fase (descritas en este capítulo).

Dado este contexto para estudiar análisis de sistemas, ahora podemos explorar las fases y tareas del análisis de sistemas.

## Fase de definición de alcance

Recuerde del capítulo 3 que la *fase de definición de alcance* es la primera del proceso de desarrollo de sistemas clásico. En otras metodologías esto podría ser llamado *fase de investigación preliminar*, *fase de estudio inicial*, *fase de investigación* o *fase de planeación*. La fase de definición de alcance responde la pregunta, “¿vale la pena realizar este proyecto?”. Para responder esta pregunta, debemos definir el alcance del proyecto y los problemas percibidos, oportunidades y directrices que dispararon el proyecto. Supongamos que el problema *se considera* digno de investigarse, la fase de definición de alcance debe también establecer el plan de proyecto en términos de escala, estrategia de desarrollo, programación, requerimientos de recursos y presupuesto.<sup>1</sup>

El contexto para la fase de definición de alcance aparece en un rectángulo punteado en la figura 4.5. Nótese que la fase de definición de alcance tiene como prioridad el punto de vista del PROPIETARIO DEL SISTEMA del ya existente y los problemas y oportunidades que dispararon el interés. Los propietarios de sistemas tienden a estar preocupados por la imagen general, no por los detalles. Más aún, ellos determinan si los recursos pueden y serán comprometidos con el proyecto.

En la figura 4.6 se encuentra el primero de cinco diagramas de tareas que presentaremos en este capítulo para dar un vistazo más de cerca a cada fase de análisis de sistemas. Un *diagrama de tareas* muestra el trabajo (= tareas) que debe ser realizado para completar una fase. Nuestros diagramas de tareas no indican una metodología específica, pero describiremos en los párrafos anexos los métodos, herramientas y técnicas que usted podría considerar para cada tarea. En la figura 4.6 se muestran las tareas requeridas para la fase de definición de alcance. Es importante recordar que estos diagramas de tareas son sólo plantillas. El equipo de proyecto y el administrador del proyecto pueden expandir o alterar estas plantillas para reflejar las necesidades únicas de cualquier proyecto dado.

Como se muestra en la figura 4.6, el producto final de la fase de investigación preliminar es el cumplimiento de una CARTA DE DEFINICIÓN DEL PROYECTO. (Esos importantes productos están indicados en cada diagrama de tareas en letras mayúsculas.) Una carta de definición de proyecto define el alcance de proyecto, el plan, la metodología, los estándares y demás. El completar la carta de definición de proyecto representa el primer hito en un proyecto.

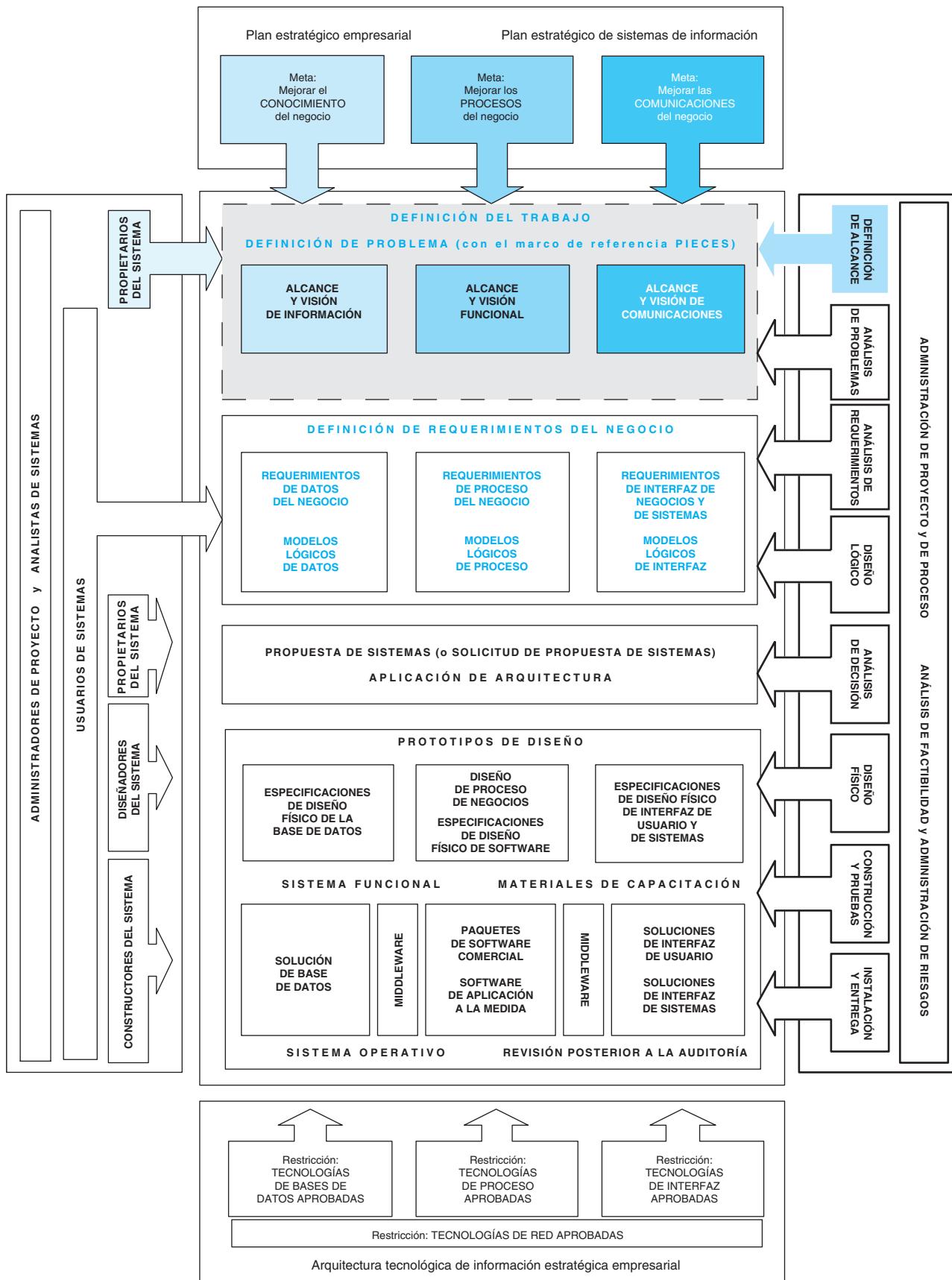
Se pretende que la fase de definición de alcance sea rápida. La fase completa no debe exceder dos o tres días para la mayoría de los proyectos. En general, la fase incluye las siguientes tareas:

- 1.1 Identificar problemas y oportunidades básicos.
- 1.2 Negociar alcance base.
- 1.3 Considerar el valor del proyecto base.
- 1.4 Desarrollar un programa y presupuesto iniciales.
- 1.5 Comunicar el plan de proyecto.

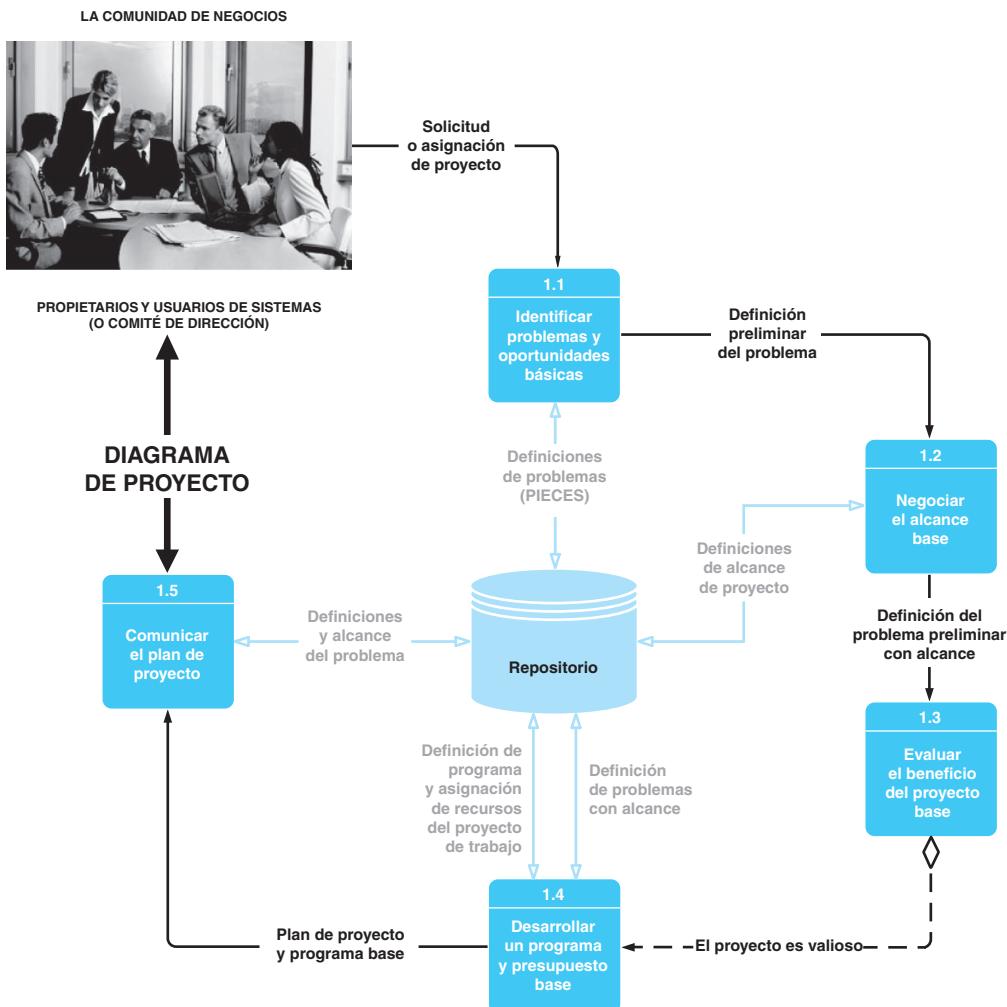
Ahora analicemos cada una de estas tareas con mayor detalle.

<sup>1</sup> Estos elementos de planeación forman parte del proceso utilizado por los administradores de proyecto para desarrollar un plan de proyecto.\*

\* Nota del R.T.: Para el lector interesado, se recomienda consultar un libro sobre administración de proyectos, como el PMBOK del PMI.



**FIGURA 4.5** El contexto de la fase de definición de alcance del análisis de sistemas

**FIGURA 4.6**

Tareas para la fase de definición de alcance del análisis de sistemas

### > Tarea 1.1: Identificar problemas y oportunidades básicas

Una de las tareas más importantes de la fase de definición de **alcance** es establecer una base inicial de los problemas, oportunidades o directrices que dispararon el proyecto. Cada problema, oportunidad y directriz está evaluado con relación a la urgencia, visibilidad, beneficios tangibles y prioridad. Cualquier análisis adicional y detallado no es relevante en esta etapa del proyecto. Sin embargo, puede ser útil listar cualquier restricción (limitación) del proyecto, como fechas límites, presupuestos máximos o arquitectura tecnológica.

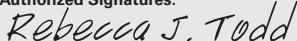
Por lo general un analista de sistemas con mucha experiencia o un administrador de proyecto dirige esta tarea. La mayoría de los otros participantes se clasifican en general como **PROPIETARIOS DEL SISTEMA**. Esto incluye a los patrocinadores ejecutivos, a los administradores de más alto nivel que pagarán y apoyarán el proyecto. También incluye administradores de todas las unidades organizacionales que pueden ser impactados por el sistema y es posible que incluya a los administradores de sistemas de información. Los **USUARIOS DE SISTEMAS**, **DISEÑADORES DE SISTEMAS** y **CONSTRUCTORES DE SISTEMAS** por lo general no participan en esta tarea.

Como se muestra en la figura 4.6, una **SOLICITUD DEL PROYECTO O ASIGNACIÓN** dispara la tarea. Este disparador puede tomar una o varias formas alternativas. Puede ser tan simple como un memorando de autoridad de un comité de dirección de sistemas de información. O puede ser un memorando de un equipo o una unidad de negocios que solicita un desarrollo de sistemas. Algunas organizaciones requieren que todas las solicitudes de proyectos sean remitidas en formato de solicitud de servicio, como el de la figura 4.7.

**alcance** Límites de un proyecto: las áreas de un negocio que el proyecto podría atender (o no).

**FIGURA 4.7**

Solicitud de servicios de sistemas

<b>SoundStage Entertainment Club</b> <i>Information System Services</i> Phone: 494-0666 Fax: 494-0999 Internet: <a href="http://www.soundstage.com">http://www.soundstage.com</a> Intranet: <a href="http://www.soundstage.com/iss">http://www.soundstage.com/iss</a>		<b>REQUEST FOR INFORMATION SYSTEM SERVICES</b>																			
<b>DATE OF REQUEST</b> <table border="1" style="width: 100%;"> <tr> <td style="padding: 2px;">January 9, 2003</td> <td style="padding: 2px;"><b>SERVICE REQUESTED FOR DEPARTMENT(S)</b></td> </tr> <tr> <td colspan="2" style="padding: 2px; text-align: center;">Member Services, Warehouse, Shipping</td> </tr> </table>		January 9, 2003	<b>SERVICE REQUESTED FOR DEPARTMENT(S)</b>	Member Services, Warehouse, Shipping																	
January 9, 2003	<b>SERVICE REQUESTED FOR DEPARTMENT(S)</b>																				
Member Services, Warehouse, Shipping																					
<b>SUBMITTED BY (key user contact)</b> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%;">Name</td> <td>Sarah Hartman</td> <td style="width: 50%;">EXECUTIVE SPONSOR (funding authority)</td> </tr> <tr> <td>Title</td> <td>Business Analyst, Member Services</td> <td>Name</td> <td>Galen Kirkhoff</td> </tr> <tr> <td>Office</td> <td>B035</td> <td>Title</td> <td>Vice President, Member Services</td> </tr> <tr> <td>Phone</td> <td>494-0867</td> <td>Office</td> <td>G242</td> </tr> <tr> <td></td> <td></td> <td>Phone</td> <td>494-1242</td> </tr> </table>		Name	Sarah Hartman	EXECUTIVE SPONSOR (funding authority)	Title	Business Analyst, Member Services	Name	Galen Kirkhoff	Office	B035	Title	Vice President, Member Services	Phone	494-0867	Office	G242			Phone	494-1242	
Name	Sarah Hartman	EXECUTIVE SPONSOR (funding authority)																			
Title	Business Analyst, Member Services	Name	Galen Kirkhoff																		
Office	B035	Title	Vice President, Member Services																		
Phone	494-0867	Office	G242																		
		Phone	494-1242																		
<b>TYPE OF SERVICE REQUESTED:</b> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%; padding-left: 10px;"> <input type="checkbox"/> Information Strategy Planning  <input checked="" type="checkbox"/> Business Process Analysis and Redesign  <input checked="" type="checkbox"/> New Application Development  <input type="checkbox"/> Other (please specify) _____         </td> <td style="width: 50%; padding-left: 10px;"> <input type="checkbox"/> Existing Application Enhancement  <input type="checkbox"/> Existing Application Maintenance (problem fix)  <input type="checkbox"/> Not Sure         </td> </tr> </table>			<input type="checkbox"/> Information Strategy Planning <input checked="" type="checkbox"/> Business Process Analysis and Redesign <input checked="" type="checkbox"/> New Application Development <input type="checkbox"/> Other (please specify) _____	<input type="checkbox"/> Existing Application Enhancement <input type="checkbox"/> Existing Application Maintenance (problem fix) <input type="checkbox"/> Not Sure																	
<input type="checkbox"/> Information Strategy Planning <input checked="" type="checkbox"/> Business Process Analysis and Redesign <input checked="" type="checkbox"/> New Application Development <input type="checkbox"/> Other (please specify) _____	<input type="checkbox"/> Existing Application Enhancement <input type="checkbox"/> Existing Application Maintenance (problem fix) <input type="checkbox"/> Not Sure																				
<b>BRIEF STATEMENT OF PROBLEM, OPPORTUNITY, OR DIRECTIVE</b> (attach additional documentation as necessary) <p>The information strategy planning group has targeted member services, marketing, and order fulfillment (inclusive of shipping) for business process redesign and integrated application development. Currently serviced by separate information systems, these areas are not well integrated to maximize efficient order services to our members. The current systems are not adaptable to our rapidly changing products and services. In some cases, separate systems exist for similar products and services. Some of these systems were inherited through mergers that expanded our products and services. There also exist several marketing opportunities to increase our presence to our members. One example includes Internet commerce services. Finally, the automatic identification system being developed for the warehouse must fully interoperate with member services.</p>																					
<b>BRIEF STATEMENT OF EXPECTED SOLUTION</b> <p>We envision completely new and streamlined business processes that minimize the response time to member orders for products and services. An order shall not be considered fulfilled until it has been received by the member. The new system should provide for expanded club and member flexibility and adaptability of basic business products and services.</p> <p>We envision a system that extends to the desktop computers of both employees and members, with appropriate shared services provided across the network, consistent with the ISS distributed architecture. This is consistent with strategic plans to retire the AS/400 central computer and replace it with servers.</p>																					
<b>ACTION (ISS Office Use Only)</b> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 50%; padding-left: 10px;"> <input type="checkbox"/> Feasibility assessment approved  <input checked="" type="checkbox"/> Feasibility assessment waived   <input type="checkbox"/> Request delayed  <input type="checkbox"/> Request rejected         </td> <td style="width: 50%; padding-left: 10px;">           Assigned to <u>Sandra Shepherd</u>            Approved Budget \$ <u>450,000</u>            Start Date <u>ASAP</u> Deadline <u>ASAP</u>            Backlogged until date: _____            Reason: _____         </td> </tr> </table>			<input type="checkbox"/> Feasibility assessment approved <input checked="" type="checkbox"/> Feasibility assessment waived  <input type="checkbox"/> Request delayed <input type="checkbox"/> Request rejected	Assigned to <u>Sandra Shepherd</u> Approved Budget \$ <u>450,000</u> Start Date <u>ASAP</u> Deadline <u>ASAP</u> Backlogged until date: _____ Reason: _____																	
<input type="checkbox"/> Feasibility assessment approved <input checked="" type="checkbox"/> Feasibility assessment waived  <input type="checkbox"/> Request delayed <input type="checkbox"/> Request rejected	Assigned to <u>Sandra Shepherd</u> Approved Budget \$ <u>450,000</u> Start Date <u>ASAP</u> Deadline <u>ASAP</u> Backlogged until date: _____ Reason: _____																				
<b>Authorized Signatures:</b>  <u>Rebecca J. Todd</u> Chair, ISS Executive Steering Body																					
 <u>Galen Kirkhoff</u> Project Executive Sponsor																					
FORM ISS-100-RFSS (Last revised December, 1999)																					

El producto fundamental de esta tarea, la DEFINICIÓN PRELIMINAR DEL PROBLEMA, consiste en problemas, oportunidades y directrices que fueron identificadas. Las DEFINICIONES DE PROBLEMAS se almacenan en el repositorio para uso posterior en el proyecto. La figura 4.8 es un documento muestra que resume problemas, oportunidades y directrices en términos de:

- **Urgencia.** ¿En qué tiempo debe ser resuelto el problema o debe ser realizada la oportunidad o directriz? Una escala de calificación puede desarrollarse para responder de manera consistente a esta pregunta.
- **Visibilidad.** ¿A qué grado una solución o un sistema nuevo debe ser visible para los clientes o la administración ejecutiva? De nuevo, podría desarrollarse una escala de calificación para las respuestas.
- **Beneficios.** ¿Aproximadamente cuánto incrementaría una solución o un sistema nuevo los ingresos anuales o reduciría los costos anuales? Esto a menudo es incierto, pero si todos los participantes cooperan en esta estimación, demostrará ser bastante conservadora.

**Definición de problema**

Proyecto:	Sistema de información de servicios a miembros	Administrador de proyecto:	Sandra Shepherd
Creado por:	Sandra Shepherd	Actualizado por última vez por:	Roberto Martínez
Fecha de creación:	9 de enero de 2003	Fecha de última actualización:	15 de enero de 2003

Definiciones breves de problema, oportunidad o directriz	Urgencia	Visibilidad	Beneficios anuales	Prioridad o lugar	Solución propuesta
1. El tiempo de respuesta del pedido medido desde el momento de recepción, hasta la entrega del mismo ha aumentado en un promedio de 15 días.	Tan pronto como sea posible	Alta	\$175 000	2	Nuevo desarrollo
2. Las recientes adquisiciones de Club de video de monitoreo privado y pantalla de juego enfatizarán más aún los requerimientos del sistema actual.	6 meses	Mediana	75 000	2	Nuevo desarrollo
3. Actualmente, tres distintos servicios de entrada de sistemas, audio, video y divisiones de juego. Cada sistema está diseñado para tener una interfaz con un sistema de almacenes distinto; por tanto, la intención de fusionar el inventario a un solo almacén se ha retrasado.	6 meses	Mediana	515 000	2	Nuevo desarrollo
4. Hay una falta general de acceso a la información administrativa y de toma de decisiones. Esto se torna difícil por la adquisición de dos sistemas de proceso de pedidos adicionales (de filtración privada y pantalla de juego).	12 meses	Baja	15 000	3	Después de que un sistema nuevo se desarrolla, es necesario proporcionar a los usuarios herramientas de reporte fáciles de aprender y utilizar
5. Actualmente existen inconsistencias de datos en los archivos de miembros y pedidos.	3 meses	Alta	35 000	1	Arreglo rápido; luego nuevo desarrollo
6. La filtración privada y los archivos de sistemas de pantalla de juegos son incompatibles con los equivalentes de SoundStage. Los problemas de datos de negocios incluyen inconsistencia de datos y falta de controles de edición de entrada.	6 meses	Mediana	Desconocidos	2	Nuevo desarrollo. Cuantificación adicional de beneficio podría incrementar la urgencia
7. Hay una oportunidad de abrir sistemas de pedidos en el Internet, pero la seguridad y el control son un tema.	12 meses	Baja	Desconocidos	4	Versión futura de un sistema recién desarrollado
8. El sistema de entrada de pedidos actual es incompatible con el siguiente sistema de identificación automática (código de barras) que se desarrolla para el almacén.	3 meses	Alta	65 000	1	Arreglo rápido; luego nuevo desarrollo

**FIGURA 4.8** Muestra de definiciones de problemas

- *Prioridad.* Con base en las respuestas anteriores, ¿cuáles son las prioridades acordadas entre todos para cada problema, oportunidad o directriz? Si el presupuesto o el programa se vuelven un problema, estas prioridades ayudarán a ajustar el alcance del proyecto.
- *Soluciones posibles (OPT).* En esta etapa inicial del proyecto, las soluciones posibles se expresan mejor en términos simples como *a)* salir de ello bastante bien, *b)* utilizar un arreglo rápido, *c)* hacer una mejora de simple a moderada del sistema existente, *d)* redesignar el sistema existente o *e)* diseñar un nuevo sistema. Los participantes listados arriba para esta tarea son los adecuados para un análisis de alto nivel de estas opciones.

El marco de referencia PIECES que se presentó en el capítulo 3 puede utilizarse como marco de referencia para clasificar problemas, oportunidades, directrices y restricciones. Por ejemplo, el problema 1 en la figura 4.8 podría clasificarse de acuerdo con PIECES como P.B.: Desempeño, tiempos de respuesta. (Véase la figura 3.4 en el capítulo 3.) El problema 4 en la figura 4.8 podría clasificarse como I.A.2: Información, Salidas, Falta de información necesaria.

Las técnicas básicas utilizadas para completar esta tarea incluyen búsqueda de datos y reuniones con los PROPIETARIOS DEL SISTEMA. Estas técnicas se enseñan en el capítulo 5.

### > Tarea 1.2: Negociar el alcance base

El alcance define los límites del proyecto; aquellos aspectos del negocio que serán incluidos en el proyecto y los que no. El alcance puede cambiar durante el proyecto; sin embargo, el plan de proyecto inicial debe establecer el alcance preliminar o base. En consecuencia, si el alcance cambia en forma significativa, todas las partes tendrán una mejor apreciación de por qué cambian también el presupuesto y el programa. Esta tarea puede ocurrir en paralelo con la tarea anterior.

Una vez más, un analista de sistemas experto o un administrador de proyecto por lo general dirigen esta tarea. La mayor parte del resto de los participantes se clasifican de manera colectiva como PROPIETARIOS DEL SISTEMA. Esto incluye al patrocinador ejecutivo, los gerentes de todas las unidades organizacionales que pueden tener impactos por el sistema y tal vez los gerentes de sistemas de información. Los USUARIOS DE SISTEMAS, DISEÑADORES DE SISTEMAS y CONSTRUCTORES DE SISTEMAS en general no participan en esta tarea.

Como se muestra en la figura 4.6, esta tarea utiliza la DEFINICIÓN DE PROBLEMA PRELIMINAR producida por la tarea anterior. Debe tener sentido que esos problemas, oportunidades y directrices forman la base para definir el alcance. Las DEFINICIONES DEL ALCANCE DEL PROYECTO se suman al repositorio para su uso posterior. Estas definiciones también se documentan de manera formal como el producto de la tarea, DEFINICIÓN DEL PROBLEMA PRELIMINAR CON ALCANCE.

El alcance puede definirse con facilidad dentro del contexto de los componentes de su sistema de información. Por ejemplo, el alcance de un proyecto puede describirse en términos de:

- ¿Qué tipos de DATOS describen al sistema que se estudia? Por ejemplo, un sistema de información de ventas puede requerir datos acerca de cosas como CLIENTES, PEDIDOS, PRODUCTOS y REPRESENTANTES DE VENTAS.
- ¿Qué PROCESOS de negocios se incluyen en el sistema que se estudia? Por ejemplo, un sistema de información de ventas puede incluir procesos de negocios para ADMINISTRACIÓN DE CATÁLOGO, ADMINISTRACIÓN DE CLIENTES, INGRESO DE PEDIDOS, SATISFACCIÓN DE PEDIDOS, ADMINISTRACIÓN DE PEDIDOS y ADMINISTRACIÓN DE LA RELACIÓN CON LOS CLIENTES.
- ¿Cómo debe ser la INTERFAZ del sistema con los usuarios, ubicaciones y los demás sistemas? Por ejemplo, las interfaces potenciales de un sistema de información de ventas podría incluir CLIENTES, REPRESENTANTES DE VENTAS, EMPLEADOS Y GERENTES DE VENTAS, OFICINAS REGIONALES DE VENTAS y LOS SISTEMAS DE CUENTAS POR COBRAR y SISTEMAS DE INFORMACIÓN DE CONTROL DE INVENTARIOS.

Nótese que cada definición de alcance puede ser descrita como una simple lista. No “definimos” de manera necesaria los productos en la lista. Ni tampoco estamos muy preocupados por un análisis de requerimientos preciso. Y en definitiva, no nos interesan los pasos que toman mucho tiempo como la elaboración de modelos o de prototipos.

Una vez más, las técnicas primarias utilizadas para completar esta tarea son la identificación de hechos y las reuniones. Muchos analistas prefieren combinar esta tarea con la anterior y la siguiente para realizarlas dentro de una sola reunión.

### > Tarea 1.3: Evaluar el beneficio del proyecto base

Aquí es donde respondemos la pregunta: “¿vale la pena trabajar en este proyecto?”. En esta etapa inicial del proyecto, la pregunta en realidad podría reducirse a hacer una “mejor suposición”. ¿Solucionar los problemas, explotar las oportunidades o satisfacer las directrices devolverá el valor suficiente para superar los costos en los que incurriríremos para desarrollar este sistema? Es imposible hacer un adecuado análisis de factibilidad con base en los hechos limitados que hemos reunido hasta el momento.

Una vez más, un analista de sistemas experto o un administrador de proyecto, por lo general dirigen esta tarea. Pero los PROPIETARIOS DEL SISTEMA, incluidos el patrocinador ejecutivo, los gerentes de unidades del negocio y los del sistema de información, deben tomar la decisión.

Como se muestra en la figura 4.6, la DEFINICIÓN PRELIMINAR DEL PROBLEMA CON EL ALCANCE dispara la tarea. Esto proporciona el nivel de información requerido para esta evaluación preliminar del valor. No hay un producto físico distinto a la decisión de SIGUE O NO SIGUE. En realidad hay varias decisiones alternativas. El proyecto puede aprobarse o cancelarse y el alcance del proyecto puede renegociarse (aumentar o disminuir). Es evidente que las tareas restantes en la fase de investigación preliminar son necesarias sólo si el proyecto ha sido considerado valioso y si tiene la aprobación para continuar.

### > Tarea 1.4: Desarrollar un programa y presupuesto base

Si el proyecto se ha considerado como valioso para continuar, ahora podemos planear la extensión del mismo. El proyecto inicial debe consistir al menos de lo siguiente:

- Un plan maestro preliminar que incluya programa y asignación de recursos para el proyecto completo. Este plan será actualizado al final de cada fase del proyecto. A veces se llama *plan base (baseline plan)*.
- Un plan y programa detallado para completar la siguiente fase del proyecto (la de análisis del problema).

La tarea es responsabilidad del *administrador del proyecto*. A la mayor parte de los administradores de proyectos les resulta útil incluir lo más posible del equipo del proyecto, incluidos los PROPIETARIOS DE SISTEMAS, USUARIOS, DISEÑADORES Y CONSTRUCTORES. Como se mostró en la figura 4.6, esta tarea se dispara por la decisión de SIGUE O NO SIGUE para continuar el proyecto. Esta decisión representa un acuerdo consensuado acerca del alcance, los problemas, las oportunidades, las directrices y el valor del proyecto. (Este “valor” debe aún ser presentado y aprobado.) Las DEFINICIONES (ACOTADAS) DEL PROBLEMA SON la entrada fundamental del repositorio. El producto de esta tarea es EL PLAN Y PROGRAMA DEL PROYECTO BASE. La DEFINICIÓN DEL TRABAJO, EL PROGRAMA DEL PROYECTO Y ASIGNACIÓN DE RECURSOS también se agregan al repositorio para una vigilancia continua y actualización, según resulte apropiado. El programa y los recursos se mantienen en forma normal en el repositorio como un archivo de software administrativo.

En la actualidad, las técnicas utilizadas para crear un plan de proyecto son soportadas por software de administración de proyectos como Microsoft *Project*.\*

### > Tarea 1.5: Comunicar el plan del proyecto

En la mayoría de las organizaciones, hay más proyectos potenciales que recursos para dar fondos y personal a esos proyectos. A menos que nuestro proyecto haya sido predeeterminado a ser de la más alta prioridad (por algún tipo de decisión táctica o proceso de planeación estratégica), debe ser presentado y defendido frente a un **comité de dirección** para su aprobación. La mayoría de las organizaciones utilizan un comité de dirección para aprobar y vigilar los proyectos y su progreso. La mayor parte de los integrantes de cualquier comité de dirección debe estar constituida por ejecutivos o administradores que no formen parte del área de los sistemas de información. Muchas organizaciones designan

---

**comité de dirección** Un comité de gerentes ejecutivos de negocios y sistemas que estudia y jerarquiza propuestas de proyectos que compiten entre sí, con el fin de determinar cuáles generarán más valor para la organización, de las cuales, algunas se aprobarán para que continúe el desarrollo de sistemas. También llamado *comité de dirección*.

\*Nota del R.T.: Al lector interesado en este tema se le recomienda consultar el PMBOK del PMI.

vicepresidentes para formar parte del comité de dirección. Otras organizaciones asignan el personal que reporta en forma directa a los vicepresidentes del comité de dirección. Algunas organizaciones utilizan dos comités de dirección, uno para vicepresidentes y uno para su personal directo. Los gerentes de sistemas de información trabajan en el comité de dirección sólo para responder preguntas y para comunicar prioridades de vuelta a los desarrolladores y a los administradores de proyecto.

Sin importar si un proyecto requiere o no de la aprobación de un comité de dirección, es de igual importancia lanzar y comunicar de manera formal el proyecto, las metas y el programa a la comunidad entera del negocio. Abrir las líneas de la comunicación es un paso importante para la investigación preliminar. Por esta razón, recomendamos las “mejores prácticas” para conducir un *evento de proyecto de salida (kickoff)* y crear un *sitio Web de proyecto de intranet*. La reunión de salida del proyecto está abierta a la comunidad entera de negocios, y no sólo a las unidades de negocios afectadas y al equipo de proyecto. El sitio Web de proyecto de intranet establece un portal de comunidad para todas las noticias no sensibles y la documentación relacionada con el proyecto.

De manera ideal, el patrocinador ejecutivo debe facilitar en forma conjunta la tarea con el administrador de proyecto elegido. La visibilidad del patrocinador ejecutivo establece una credibilidad instantánea y una prioridad para todos los que participan en la reunión de salida. Otros participantes en la reunión de salida deben incluir el equipo de proyecto completo, incluidos los PROPIETARIOS, USUARIOS, ANALISTAS, DISEÑADORES y CONSTRUCTORES DE SISTEMAS asignados. Es preferible que la reunión de salida esté abierta a cualquier miembro del personal interesado de la comunidad de negocios. Esto crea conocimiento y hace un consenso entre los involucrados al tiempo que reduce tanto el volumen como las consecuencias del rumor y de la mala información. Para el componente de intranet, un Webmaster o un autor Web debe ser asignado al equipo de proyecto.

Como se muestra en la figura 4.6, esta tarea se dispara por el cumplimiento del PLAN Y PROGRAMA DEL PROYECTO BASE. LAS DEFINICIONES DE PROBLEMA y el ALCANCE están disponibles en el repositorio. El producto es el ESQUEMA DEL PROYECTO y éste, por lo general, es un documento. Incluye diversos elementos que definen el proyecto en términos de participantes, problemas, oportunidades y directrices; alcance, metodología, definición del trabajo que se debe completar; productos, estándares de calidad, programa y presupuesto. El esquema del proyecto debe agregarse al sitio Web del proyecto para que todos lo veamos. Los elementos del esquema del proyecto también pueden ser presentados como diapositivas y boletines (por medio de software como *PowerPoint* de Microsoft) para incluirlos en el suceso de inicio del proyecto.

Las habilidades interpersonales y la comunicación son las claves para esta tarea. Éstas incluyen principios de persuasión, compra-venta, creación de contratos y hablar en público.

Con esto terminamos nuestro análisis de la fase de definición de alcance. En esta fase, los participantes podrían decidir que el proyecto no vale la pena para proponerse. También es posible que el comité de dirección pueda decidir que otros proyectos son más importantes. O el patrocinador ejecutivo podría no avalar el proyecto. En cada uno de estos casos, el proyecto se termina. Poco tiempo y esfuerzo se han utilizado. Por otro lado, con la bendición de todos los propietarios de sistemas y del comité de dirección, el proyecto podría continuar a la fase de análisis del problema.

## Fase de análisis del problema

Hay un viejo dicho que dice: “no trates de arreglarlo si no lo entiendes”. Esta declaración describe en forma adecuada la *fase de análisis del problema* del análisis de sistemas. Siempre hay un sistema actual o existente, sin importar a qué grado esté automatizado con tecnología de información. La fase de análisis del problema proporciona al analista una comprensión más completa de los problemas, oportunidades y/o directrices que dispararon el proyecto. La fase de análisis del problema responde a las preguntas, “¿vale la pena resolver estos problemas?” y “¿vale la pena construir un nuevo sistema?”. En otras metodologías, la fase de análisis de problemas puede ser conocida como *fase de estudio, estudio del sistema actual, fase de investigación detallada o fase de análisis de factibilidad*.

¿Alguna vez se puede pasar por alto la fase de análisis del problema? ¡Rara vez! Casi siempre se necesita un nivel de comprensión del sistema actual. Pero puede haber razo-

nes para acelerar la fase del análisis del problema. Primero, si el proyecto fue disparado por un plan estratégico o táctico, es probable que el valor del proyecto no esté en duda, la fase del análisis de problema se reduciría a una comprensión del sistema actual, no a analizarlo. Segundo, un proyecto puede ser iniciado por una directriz (tal como el cumplimiento con una instrucción y vencimiento gubernamental). De nuevo, en este caso el valor del proyecto no está en duda. Por último, algunas metodologías y organizaciones de manera deliberada consolidan el análisis del problema y las fases de análisis de requerimientos para acelerar el análisis de sistemas.

La meta de la fase del análisis del problema es estudiar y comprender el dominio del problema lo bastante bien para analizar a fondo sus problemas, oportunidades y restricciones. Algunas metodologías definen una comprensión muy detallada del sistema actual y lo documentan con sumo detalle mediante modelos de sistemas como diagramas de flujo de datos. En la actualidad, excepto cuando los procesos de negocios deben ser rediseñados, el esfuerzo requerido y el valor agregado por dichos modelos detallados se cuestionan y por lo general se ignoran. Por tanto, la versión actual de nuestra metodología hipotética *FAST* define un modelado de sistemas suficientemente amplio para refinar nuestra comprensión del alcance del proyecto, la definición del problema y la definición de un vocabulario común para el sistema.

El contexto para la fase del análisis del problema está sombreado en la figura 4.9. Nótese que la fase del análisis del problema trata en su mayor parte con los puntos de vista de los PROPIETARIOS y USUARIOS DEL SISTEMA acerca del sistema existente. Nótese que construimos sobre las listas creadas en la fase de investigación preliminar para analizar los componentes del CONOCIMIENTO, PROCESO y COMUNICACIONES del sistema existente. También nótese que determinamos la elaboración del mínimo modelo de sistema. Podemos aún utilizar el marco de referencia PIECES para analizar problemas, causas y efectos en cada componente del sistema.

La figura 4.10 es el diagrama de tareas para la fase de análisis de problemas. El producto e hito de la fase final es producir OBJETIVOS DE MEJORA DE SISTEMAS que aborden los problemas, oportunidades y directrices. Según se conozca el tamaño del sistema, su complejidad y el grado en que el proyecto es valioso, las tareas ilustradas pueden consumir de una a seis semanas. La mayoría de estas tareas pueden ser aceleradas por sesiones estilo JRP. La fase de análisis de problemas por lo general incluye las siguientes tareas:

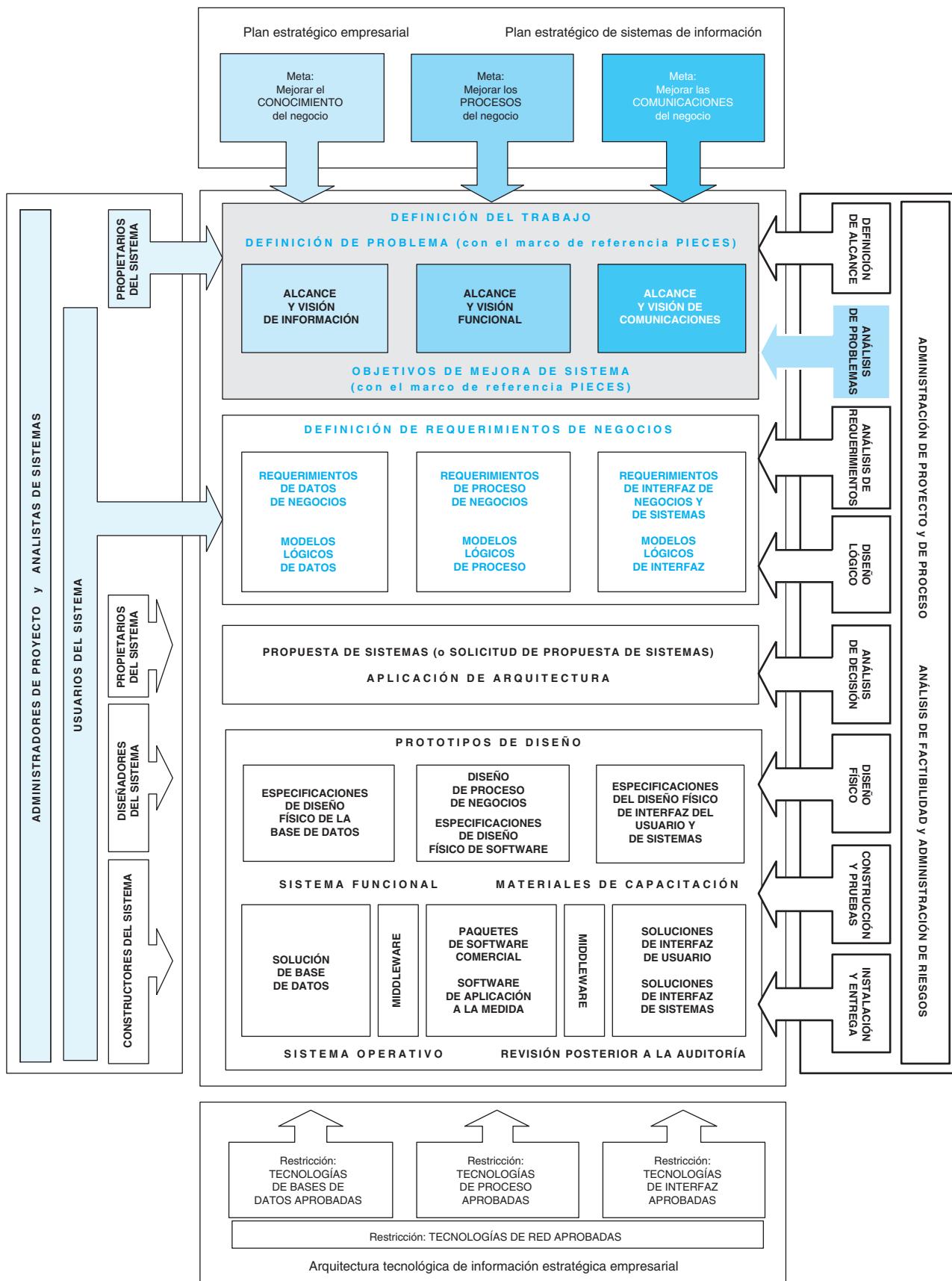
- 2.1 Entender el dominio del problema.
- 2.2 Analizar problemas y oportunidades.
- 2.3 Analizar procesos de negocios.
- 2.4 Establecer objetivos de mejora del sistema.
- 2.5 Actualizar o refinar el plan del proyecto.
- 2.6 Comunicar resultados y recomendaciones.

Ahora analicemos cada una de estas tareas con mayor detalle.

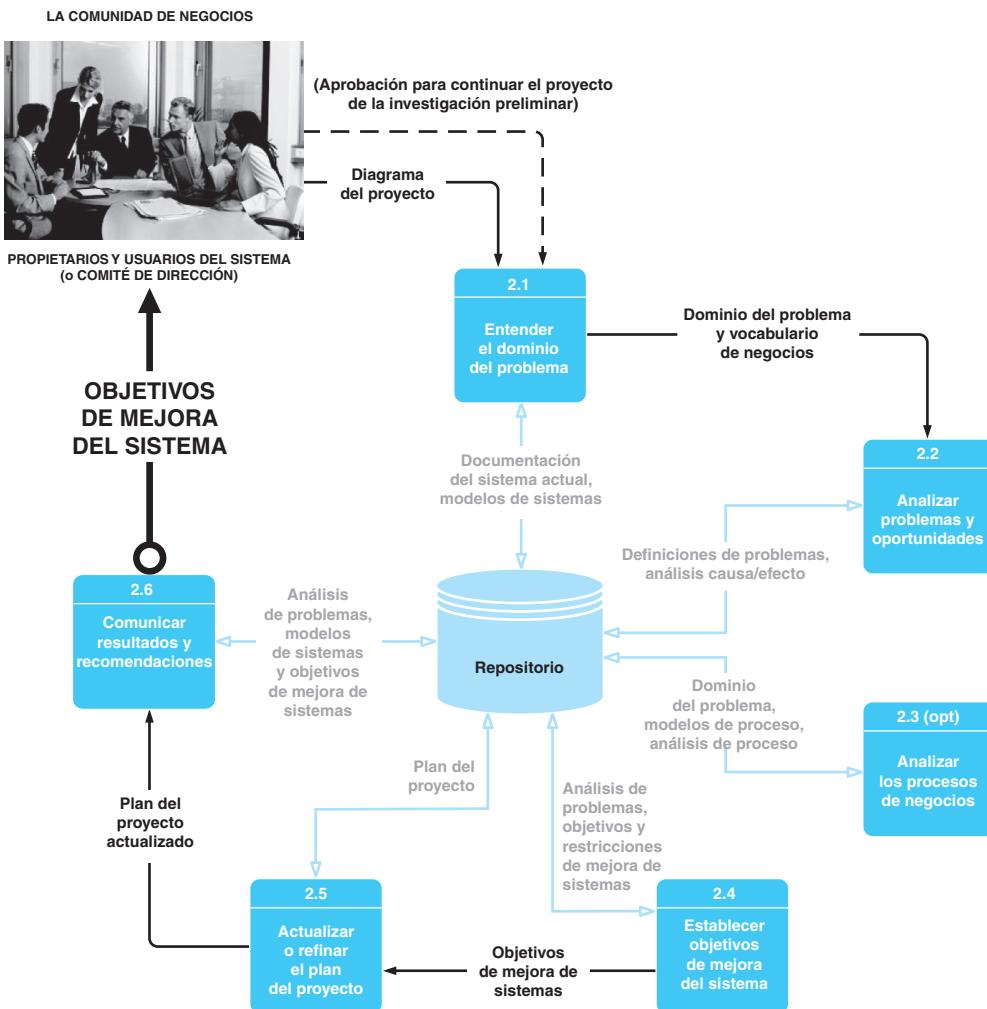
### > Tarea 2.1: Entender el dominio del problema

Durante la fase de análisis de problema, el *equipo* al inicio intenta conocer sobre el sistema actual. Cada PROPIETARIO, USUARIO y ANALISTA DE SISTEMAS aporta un nivel diferente de comprensión al sistema —detalle, vocabulario, percepciones y opiniones diferentes—. Un estudio bien conducido puede demostrar ser revelador para todas las partes, incluida la administración y los usuarios propios del sistema. Es importante estudiar y *entender el dominio del problema*, ese dominio en el que existen los problemas, oportunidades, directrices y restricciones del negocio.

Esta tarea será dirigida por el administrador del proyecto pero será facilitada por el analista de sistemas. Es común que un individuo desempeñe ambos papeles (como lo hace Sandra en el caso de SoundStage). Otros ANALISTAS DE SISTEMAS pueden también participar ya que realizan entrevistas, participan en juntas y documentan los resultados. Un estudio completo debe incluir PROPIETARIOS y USUARIOS DE SISTEMAS representativos de todas las unidades de negocios que serán apoyados o impactados por el sistema y el proyecto. Es muy importante que se incluyan suficientes usuarios para abarcar el alcance total del sistema que se estudia. En algunas organizaciones, uno o más usuarios experimentados son “prestados” al proyecto de tiempo completo como *analistas de sistemas*; sin embargo,



**FIGURA 4.9** Contexto de la fase de análisis de problemas del análisis de sistemas

**FIGURA 4.10**

Tareas para la fase de análisis de problemas del análisis de sistemas

es raro que cualquier usuario pueda representar en su totalidad los intereses de todos los usuarios. Sin embargo, los analistas de sistemas pueden fungir como facilitadores para hacer que participen las personas correctas y sostener una comunicación eficaz de vuelta con las unidades del negocio y la administración. Los DISEÑADORES y los CONSTRUCTORES DE SISTEMAS rara vez participan en esta tarea a menos que sean entrevistados para determinar cualquier limitación técnica del sistema actual.

En la figura 4.10 esta tarea se dispara por APROBACIÓN PARA CONTINUAR EL PROYECTO; desde la fase de definición de alcance. (La línea punteada indica que esta aprobación es un suceso o disparador, no un flujo de datos o información.) La aprobación viene de los PROPIETARIOS DEL SISTEMA o de un comité de dirección. La entrada de información fundamental viene de los PROPIETARIOS DEL SISTEMA y cualquier DOCUMENTACIÓN DE SISTEMA ACTUAL que pueda existir en el repositorio y en las bibliotecas del programa para el sistema actual. La documentación del sistema actual no siempre existe. Y cuando existe, debe ser revisada con sumo cuidado para ver su actualidad; es notorio que la mayor parte de dicha documentación es obsoleta debido a que los programadores y analistas no siempre son diligentes para actualizarla cuando ocurren cambios a lo largo de la vida de un sistema.

Los productos de esta tarea son una comprensión del DOMINIO DEL PROBLEMA y del VOCABULARIO DE NEGOCIOS. Su comprensión del dominio del problema existente debe ser documentada para que pueda verificarse que se entiende bien. Hay diversas formas de documentar el dominio del problema. Es cierto, dibujar MODELOS DEL SISTEMA del que hay en la actualidad puede ayudar, pero eso puede llevar a un fenómeno llamado “parálisis de análisis” en el que el deseo de producir modelos perfectos se vuelve contraproducente para el cronograma. Otro

método podría ser utilizar los componentes de su sistema de información como marco de referencia para listar y definir el dominio del sistema:

- CONOCIMIENTO. Liste todas las “cosas” acerca de las cuales el sistema en la actualidad almacena datos (en archivos, bases de datos, formatos, etcétera). Defina cada cosa en términos de negocios. Por ejemplo, “un PEDIDO es una transacción de negocios en la que un cliente solicita adquirir productos”.  
Además, podríamos listar todos los informes producidos por el sistema actual y describir su propósito o uso. Por ejemplo, “el informe de pedidos abiertos describe todos los pedidos que no han sido surtidos una semana después de haber sido aprobados. El informe se utiliza para iniciar una administración de la relación con los clientes a través del contacto personal”.
- PROCESOS. Defina cada suceso de negocios para el cual una respuesta de negocios (proceso) es implementada en la actualidad. Por ejemplo, “un cliente coloca un nuevo pedido” o “un cliente solicita cambios a un pedido ya realizado” o “un cliente cancela un pedido”.
- COMUNICACIONES. Defina todas las ubicaciones que el sistema actual atiende y todos los usuarios en cada una de esas locaciones. Por ejemplo, “el sistema es utilizado en la actualidad en las oficinas de ventas regionales en San Diego, Dallas, St. Louis, Indianapolis, Atlanta y Manhattan. Cada oficina de ventas regionales tiene un gerente de ventas, un asistente del gerente de ventas, asistente administrativo y de 5 a 10 empleados de ventas, todos los cuales utilizan el sistema actual. Cada región también es la base de entre 5 y 30 representantes de ventas que viajan casi todos los días pero que cargan pedidos y otras transacciones todas las tardes”.

Otra faceta de las interfaces son las interfaces del sistema; es decir, aquellas que existen entre el sistema de información actual y otros sistemas de información y aplicaciones de cómputo. Éstas pueden ser listadas de manera rápida y ser descritas por el personal de sistemas de información.

Por último, la metodología de desarrollo de sistemas de la organización y el plan de proyecto determinará qué tipos y nivel de documentación son los esperados.

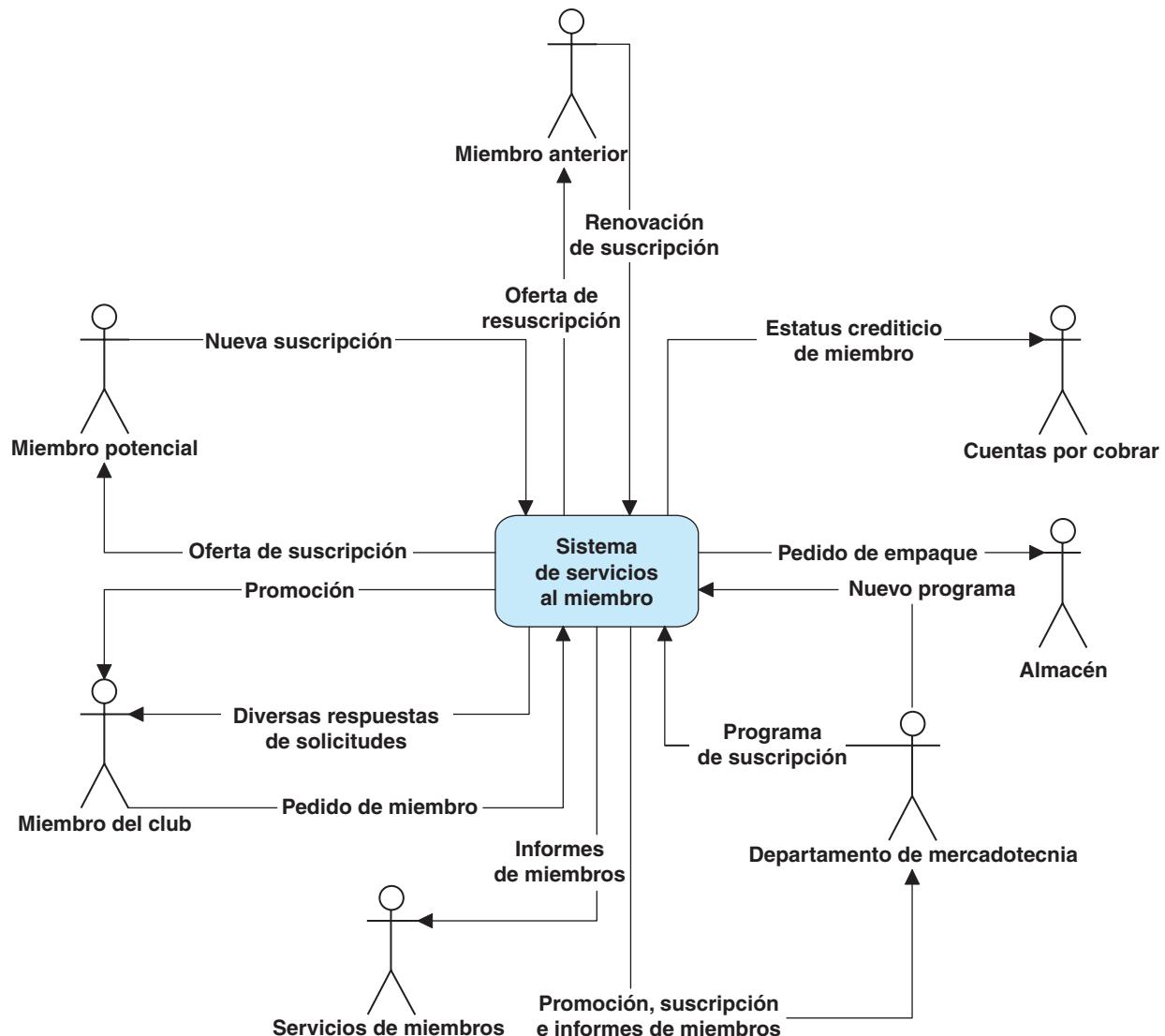
El producto de vocabulario de negocios se abrevia con demasiada frecuencia. Entender el vocabulario de negocios del sistema es una forma excelente de comprender el sistema mismo. Construye puentes para salvar la brecha de comunicación que a menudo existe o se desarrolla entre los expertos de negocios y los de tecnología.

Si elige realizar MODELOS DE SISTEMAS durante esta tarea, sugerimos que si “desea aprender *algo*, no debe tratar de aprender *todo*; al menos no en esta tarea”. Para evitar una parálisis de análisis, sugerimos que los modelos de sistemas siguientes pueden ser apropiados:

- CONOCIMIENTO. Un modelo de datos de una página es muy útil para establecer vocabulario y reglas de negocios. La elaboración de modelos de datos se enseña en el capítulo 7.
- PROCESOS. En la actualidad, es muy aceptado que un diagrama de descomposición funcional de una o dos páginas debe ser suficiente para tener la idea del procesamiento del sistema actual. La elaboración de modelos de descomposición se enseña en el capítulo 8.
- COMUNICACIONES. Un diagrama de contexto de una página o los diagramas de caso de uso son muy útiles para ilustrar las entradas y salidas de un sistema con otras organizaciones, unidades de negocios y sistemas. Los diagramas de contexto se analizan más adelante. Los diagramas de casos de uso se enseñan en el capítulo 6.

Otras técnicas y habilidades son útiles para desarrollar una comprensión de un sistema existente. Es evidente que, las técnicas de búsqueda de resultados (enseñadas en el siguiente capítulo) son críticas para aprender acerca de cualquier sistema existente. También, las técnicas de planeación conjunta de requerimientos o JRP (enseñadas en el siguiente capítulo) pueden acelerar esta tarea. Por último, la capacidad de comunicar en forma clara a los usuarios lo que ha aprendido acerca de un sistema también es crucial.

**Diagrama de contexto** El propósito de un diagrama de contexto es analizar cómo el sistema interactúa con el mundo y especificar en términos generales las entradas y salidas del sistema. Los diagramas de contexto pueden dibujarse de diversas formas. En el capítulo 8 se presenta el formato tradicional, que fue hecho como el primer paso para dibujar diagramas de flujo de datos.



**FIGURA 4.11** Diagrama de contexto

En el capítulo 6 se muestra un formato diferente para un diagrama de contexto, el que está mostrado en la figura 4.11 utiliza un método híbrido. Utiliza símbolos de casos de uso ya que éstos se vuelven una herramienta por lo general aceptada de la fase de análisis de requerimientos.

El sistema mismo se muestra como una "caja negra" en medio del diagrama. No estamos listos para ver dentro de esa caja. Por ahora, sólo queremos ver cómo todos la utilizarán. Las figuras de líneas alrededor de la parte externa del diagrama son las personas, las organizaciones y los otros sistemas de información que interactuarán con el sistema. En los casos de uso, éstos se llaman actores y así podemos nombrarlos aquí. En los diagramas de flujo de datos tradicionales, se llaman agentes externos. En los capítulos 6 y 8 aprenderá que una vez que usted mira dentro de la caja del sistema, otros factores como el tiempo o aparatos como sensores pueden también ser actores o agentes externos. Pero en un diagrama de contexto rara vez se incluyen.

Las líneas indican las entradas (flechas que apuntan hacia el sistema) proporcionadas por los actores al sistema y las salidas (flechas que apuntan hacia los actores) creadas por el sistema. Cada entrada y salida se identifica con una frase propia que la describe.

Para construir un diagrama de contexto pregunte a los usuarios a qué transacciones de negocios debe responder el sistema; estas son las entradas. También pregunte a los

usuarios qué reportes, notificaciones y demás salidas deben ser producidas por el sistema. Un sistema puede tener muchos reportes o informes que pueden sobrecargar de manera rápida el diagrama; consolídelos como sea necesario para mantener legible el diagrama. Se analizarán en forma separada durante otras fases en el proceso.

Es cierto que no podríamos construir un sistema de información a partir de un diagrama de contexto. Pero es un primer paso sólido. Desde este diagrama simple, sabemos a qué entradas debe responder el sistema y qué salidas debe producir. En otras palabras, nos ayuda a entender el dominio del problema. Veremos en el capítulo 6 cómo detectar casos de uso a partir de un diagrama de contexto. Ese será el primer paso en abrir la “caja negra”. Seguimos los principios para el desarrollo de sistemas presentados en el capítulo 2: “utilice un método de solución de problemas” y “divida y vencerá”.

### > Tarea 2.2: Analizar problemas y oportunidades

Además de aprender acerca del sistema actual, el equipo del proyecto debe trabajar con los propietarios y los usuarios de sistemas para *analizar problemas y oportunidades*. Usted puede preguntarse, “¿no se identificaron con anterioridad los problemas y oportunidades, en la fase de investigación preliminar?” Sí, así fue. Pero esos problemas iniciales pueden ser sólo síntomas de otros, tal vez problemas no tan bien conocidos o entendidos por los usuarios. Además, en realidad no hemos analizado ninguno de esos problemas en el sentido clásico.

El análisis verdadero de problemas es una habilidad difícil de dominar, especialmente para los analistas de sistemas inexpertos. La experiencia sugiere que la mayoría de los analistas de sistemas (y muchos de los propietarios y usuarios de sistemas) tratan de solucionar problemas sin analizarlos de verdad. Ellos podrían definir un problema como este: “Necesitamos...” o “Queremos...”. Al hacer esto, definen el problema en términos de una solución. Los solucionadores de problemas más eficaces han aprendido a analizar el problema antes de afirmar una posible solución. Analizan cada problema percibido para sus **causas y efectos**. En la práctica, un efecto puede ser en realidad un síntoma de un problema arraigado en forma más profunda o más básica. Ese problema también debe ser analizado para encontrar sus causas y efectos y así sucesivamente hasta que llegue el momento en que las causas y los efectos no arrojen síntomas de otros problemas. El análisis de causa y efecto lleva a una verdadera comprensión de los problemas y puede conducir a soluciones más creativas y valiosas que no eran tan evidentes.

LOS ANALISTAS DE SISTEMAS facilitan esta tarea; sin embargo, todos los PROPIETARIOS Y USUARIOS DE SISTEMAS deben participar de manera activa en el proceso del análisis de causa y efecto. Ellos son los expertos de dominio del problema. Los DISEÑADORES Y CONSTRUCTORES DE SISTEMAS por lo general no participan en este proceso a menos que sean llamados para analizar problemas técnicos que puedan existir en el sistema actual.

Como se muestra en la figura 4.10, la comprensión del equipo del DOMINIO DEL SISTEMA Y VOCABULARIO DE NEGOCIOS dispara esta tarea. Esta comprensión del dominio del problema es crucial debido a que los miembros del equipo no deben intentar analizar problemas a menos que entiendan el dominio en el que éstos ocurren. La otra entrada de información de esta tarea son las DEFINICIONES DEL PROBLEMA (de la fase de definición de alcance). Los productos de esta tarea son las DEFINICIONES DEL PROBLEMA (desde la fase de definición de alcance). Los productos de esta tarea son las DEFINICIONES DEL PROBLEMA actualizados y el ANÁLISIS DE CAUSA-EFECTO para cada problema y oportunidad. En la figura 4.12 se ilustra una forma de documentar un análisis de causa y efecto.

Una vez más, la identificación de hechos y las técnicas JRP son cruciales para esta tarea. Estas técnicas, así como el análisis de causa y efecto se enseñan en el siguiente capítulo.

### > Tarea 2.3: Analizar los procesos del negocio

Esta tarea es apropiada sólo para proyectos de *rediseño de procesos de negocios (BRP)* o proyectos de desarrollo de sistemas que se construyen sobre un significativo rediseño del proceso de negocios o que requieren de él. En dicho proyecto, al equipo se le pide examinar sus procesos de negocios en mucho mayor detalle para medir el valor agregado o sustraído por cada proceso en lo que se relacione con la organización total. El análisis de procesos de negocios puede estar influido por la política. Los propietarios y usuarios del

---

#### análisis de causa y efecto

Técnica en la que se estudian problemas para determinar sus causas y efectos.

**MATRIZ DE PROBLEMAS, OPORTUNIDADES, OBJETIVOS Y RESTRICCIONES**

Proyecto:	Sistema de información de servicios a miembros	Administrador del proyecto:	Sandra Shepherd
Creado por:	Roberto Martínez	Actualizado la última vez por:	Roberto Martínez
Fecha de creación:	21 de enero de 2003	Actualizado por última vez el:	31 de enero de 2003

ANÁLISIS DE CAUSA Y EFECTO		OBJETIVOS DE MEJORA DEL SISTEMA	
Problema u oportunidad	Causas y efectos	Objetivo del sistema	Restricción de sistema
1. El tiempo de respuesta al pedido es inaceptable.	<p>1. La entrada de pedidos ha aumentado al tiempo que el número de empleados de pedidos ha disminuido. El tiempo para procesar un solo pedido se mantiene relativamente constante.</p> <p>2. El sistema es demasiado dependiente del teclado. Muchos de los mismos valores tienen clave para la mayoría de los pedidos. El resultado neto toma en cuenta (con el sistema actual) que cada pedido toma más tiempo del requerido para procesarse.</p> <p>3. La edición de datos es procesada por una AS/400. Mientras esa computadora va saturando su capacidad, las respuestas de edición de pedidos se van retrasando. Mientras los empleados tratan de trabajar más rápido para mantenerse al paso con el volumen de pedidos, el número de errores aumenta.</p> <p>4. El almacenamiento de tarjetas perforadas nunca fue diseñado para maximizar la eficiencia de la satisfacción de órdenes (pedidos). Cuando crecieron las operaciones del almacén, los retrasos en cuanto al llenado de pedidos fueron inevitables.</p>	<p>1. Disminuir el tiempo requerido para procesar un solo pedido en 30 por ciento.</p> <p>2. Eliminar la entrada de datos en el tablero en 50 por ciento para todos los pedidos.</p> <p>3. Para los pedidos pendientes, reducir en lo posible los golpes de teclado al reemplazarlos por objetos de señalar y dar clic en la pantalla de despliegue de la computadora.</p> <p>4. Mover la edición de datos de una computadora compartida al escritorio.</p> <p>5. Reemplazar las tarjetas perforadas existentes con un sistema de comunicación entre los servicios proporcionados y el almacén.</p>	<p>1. No habrá un aumento en la fuerza de trabajo de proceso de pedidos.</p> <p>2. Cualquier sistema desarrollado debe ser compatible con el estándar de escritorio Windows 95 existente.</p> <p>3. El nuevo sistema debe ser compatible con el sistema de identificación automática ya aprobado (para el código de barras).</p>

**FIGURA 4.12** Muestra de análisis de causa y efecto

sistema por igual pueden volverse muy defensivos acerca de sus procesos de negocios existentes. Los analistas que participan deben mantener el enfoque en los procesos y no en las personas que los realizan y recordar de manera constante a todos que la meta es identificar oportunidades para un cambio de negocios fundamental que beneficiará al negocio y a todos lo que están dentro de él.

Uno o más analistas de sistemas o de negocios facilitan la tarea. De manera ideal, los ANALISTAS deben ser experimentados, capacitados o certificados en los métodos BPR. Los únicos participantes restantes deben ser PROPIETARIOS y USUARIOS DE SISTEMAS. El análisis de proceso de negocios debe evitar cualquier tentación de enfocarse en soluciones de tecno-

logía de información hasta mucho después de que los procesos de negocios han sido rediseñados para una máxima eficiencia. Algunos analistas encuentran útil asumir la existencia de "personas perfectas" y "tecnología perfecta" que pueda hacer "possible" cualquier cosa. Preguntan, "si el mundo fuera perfecto, ¿necesitaríamos ese proceso?"

Como se describe en la figura 4.10, una tarea de análisis de proceso de negocios depende sólo de cierto conocimiento del DOMINIO DEL PROBLEMA (de la tarea 2.1). Los productos de esta tarea son MODELOS DE PROCESOS y ANÁLISIS DE PROCESOS "tal como son" del negocio. Los modelos de proceso pueden parecerse mucho a diagramas de flujo de datos (figura 4.2) excepto que se escriben en forma característica para mostrar 1) el volumen de flujo de datos a través de los procesos, 2) los tiempos de respuesta de cada proceso y 3) cualquier retraso o cuello de botella que ocurre en el sistema. Los datos del análisis de proceso proporcionan información adicional tal como *a)* el costo de cada proceso, *b)* el valor agregado de cada proceso y *c)* las consecuencias de eliminar u orientar el proceso. Con base en los modelos "tal como son" y sus análisis, el equipo desarrolla modelos "como serán" que rediseñan los procesos de negocios para eliminar la redundancia y la burocracia e incrementar la eficiencia y el servicio.

Diversas técnicas son aplicables a esta tarea. Una vez más, las técnicas de identificación de hechos y las reuniones del equipo (capítulo 5) son muy útiles. También, las técnicas de elaboración de modelos de procesos (capítulo 8) son críticas para el éxito del BPR.

### > Tarea 2.4: Establecer objetivos de mejora del sistema

**objetivo** Una medición del éxito. Es algo que se espera lograr, si se tienen recursos suficientes.

**restricción** Algo que limita la flexibilidad en la definición de una solución según los objetivos que se tengan. En lo esencial, es imposible modificar las restricciones.

Dada nuestra comprensión del alcance del sistema actual, los problemas y oportunidades, ahora podemos *establecer los objetivos de mejora del sistema*. El propósito de esta tarea es establecer el criterio en contra del cual cualquier mejora al sistema será medida y para identificar cualquier restricción que pueda limitar la flexibilidad en lograr esas mejoras. El criterio para el éxito debe ser medido en términos de **objetivos**. Los objetivos representan el primer intento por establecer expectativas para cualquier sistema nuevo. Además de identificar objetivos, debemos también identificar cualquier restricción conocida. Las **restricciones** colocan limitaciones o delimitaciones para lograr los objetivos. Los vencimientos, objetivos y tecnologías requeridas son ejemplos de restricciones.

Los ANALISTAS DE SISTEMAS facilitan esta tarea. Otros participantes incluyen a los mismos PROPIETARIOS DE SISTEMAS y USUARIOS que han participado en otras tareas en esta fase de análisis de problemas. Nuevamente, no estamos aún preocupados por la tecnología; por tanto, los DISEÑADORES y los CONSTRUCTORES DE SISTEMAS no participan en esta tarea.

Esta tarea es disparada por el ANÁLISIS DE PROBLEMAS completado en las tareas 2.2 y 2.3. Para cada problema verificado y significativo, los analistas y los usuarios deben definir OBJETIVOS DE MEJORA DE SISTEMAS específicos. Deben también identificar cualquier RESTRICCIÓN que pueda limitarlos o evitar que logren los objetivos de mejora del sistema.

Los objetivos de mejora del sistema deben ser definiciones precisas y medibles del desempeño del negocio que definen las expectativas del nuevo sistema. Algunos ejemplos son:

Reducir el número de cuentas incobrables de clientes en 50 por ciento dentro del siguiente año.

Aumentar en 25 por ciento el número de solicitudes de préstamo que pueden ser procesadas en un turno de ocho horas.

Disminuir en 50 por ciento el tiempo requerido para reprogramar un lote de producción cuando una estación de trabajo tiene un mal funcionamiento.

Lo siguiente es un ejemplo de un mal objetivo:

Crear un informe de cuentas morosas.

Este es un mal objetivo porque afirma sólo un requerimiento y no un objetivo real.

Ahora vamos a replantear ese objetivo:

Reducir pérdidas de crédito en 20 por ciento a través de la identificación temprana de cuentas morosas.

Esto nos brinda una mayor flexibilidad. Sí, el informe de cuentas morosas funcionaría. Pero una investigación de la morosidad de un cliente podría proporcionar una forma todavía mejor de lograr el mismo objetivo.

Los objetivos de mejora de sistemas pueden ser acotados por restricciones identificables. Las restricciones caen dentro de cuatro categorías, como se listan a continuación (con ejemplos):

- *Programa:* El nuevo sistema debe ser operativo para el 15 de abril.
- *Costo:* El nuevo sistema no puede costar más de 350 000 dólares.
- *Tecnología:* El nuevo sistema debe estar en línea o todos los nuevos sistemas deben utilizar el sistema de administración de base de datos DB2.
- *Política:* El nuevo sistema debe utilizar técnicas de inventario de doble declinación de balance.

Las últimas dos columnas de la figura 4.12 documentan los objetivos y restricciones de mejora de un sistema típico.

### > Tarea 2.5: Actualizar o refinar el plan del proyecto

Recuerde que el alcance del proyecto es un objetivo móvil. Con base en nuestro programa y presupuesto base de la fase de definición de alcance, el alcance puede haber crecido o disminuido en tamaño y complejidad. (¡El crecimiento es mucho más común!) Ahora que nos aproximamos a la terminación de la fase del análisis del problema, debemos reevaluar el alcance del proyecto y, en consecuencia, *actualizar o refinar el plan de proyecto*.

El administrador del proyecto, en conjunto con los PROPIETARIOS DEL SISTEMA y el equipo completo del proyecto, facilitan esta tarea. Los ANALISTAS DEL SISTEMA y los PROPIETARIOS DEL SISTEMA son los individuos fundamentales en esta tarea. Los analistas y los propietarios deben considerar la posibilidad de que no todos los objetivos pueden ser satisfechos por el nuevo sistema. ¿Por qué? El nuevo sistema puede ser más grande de lo esperado y puede requerirse reducir el alcance para cumplir con una fecha límite. En este caso, el propietario del sistema clasificará los objetivos en orden de importancia. Luego, si el alcance debe ser reducido, los objetivos de mayor prioridad le dirán al analista qué es lo más importante.

Como se muestra en la figura 4.10, esta tarea se dispara como resultado de la definición de los OBJETIVOS DE MEJORA DEL SISTEMA. El PLAN DE PROYECTO inicial es otra entrada fundamental y el PLAN DE PROYECTO ACTUALIZADO es la salida principal. El plan actualizado debe ahora incluir un plan detallado que se debe seguir en la fase de análisis de requerimientos.\*

### > Tarea 2.6: Comunicar resultados y propuestas

Al igual que con la fase de definición de alcance, la fase de análisis de problema concluye con una tarea de comunicación. Debemos *comunicar resultados y propuestas* a la comunidad del negocio. Otra reunión en la que los participantes deben incluir al equipo del proyecto completo, incluyendo PROPIETARIOS, USUARIOS, ANALISTAS, DISEÑADORES y CONSTRUCTORES DE SISTEMAS asignados. Y, como siempre, la reunión debe estar abierta a cualquier empleado interesado que forme parte de la comunidad del negocio. También, si se estableció un sitio web en la intranet para el proyecto, debe haberse mantenido a lo largo de la fase de análisis del problema para asegurar la comunicación continua del progreso del proyecto.

Esta tarea se dispara por la terminación del PLAN DEL PROYECTO ACTUALIZADO. Las entradas de información incluyen los ANÁLISIS DE PROBLEMAS, cualquier MODELO DE SISTEMAS, los OBJETIVOS DE MEJORA DE SISTEMAS, el principal producto de la fase de análisis de problemas. El formato puede ser un informe, una presentación verbal o una inspección por un auditor o grupo de compañeros (llamado *grupo de revisión*). En la figura 4.13 se muestra un esquema de un informe escrito.

Las habilidades interpersonales y de comunicaciones son básicas para esta tarea. Los analistas de sistemas deben tener la capacidad de escribir un informe de negocios formal y de hacer una presentación de negocios sin entrar en temas o alternativas técnicas.

\* Nota del R.T.: Las técnicas y los pasos para mejorar el plan de proyecto se enseñan en el PMBOK del PMI.

**FIGURA 4.13** Esquema para objetivos de mejora de sistemas e informe de recomendaciones

Análisis del sistema \_\_\_\_\_ actual

- I. Resumen ejecutivo (aproximadamente dos páginas)
  - A. Resumen de recomendación
  - B. Resumen de problemas, oportunidades y directrices
  - C. Breve definición de objetivos de mejora de sistemas
  - D. Explicación breve de contenidos de informes
- II. Información de antecedentes (aproximadamente dos páginas)
  - A. Lista de entrevistas y facilitación de reuniones de grupo realizadas
  - B. Lista de otras fuentes de información que fueron explotadas
  - C. Descripción de técnicas analíticas utilizadas
- III. Panorama del sistema actual (aproximadamente cinco páginas)
  - A. Implicaciones estratégicas (si el proyecto es parte de o impacta un plan estratégico de sistemas de información existente)
  - B. Modelos del sistema actual
    1. Modelo de interfaz (muestra el alcance del proyecto)
    2. Modelo de datos (muestra el alcance del proyecto)
    3. Modelos geográficos (muestra el alcance del proyecto)
    4. Modelo de proceso (muestra sólo la descomposición funcional)
- IV. Análisis del sistema actual (aproximadamente 5-10 páginas)
  - A. Problemas de desempeño, oportunidades y análisis causa-efecto
  - B. Problemas de información, oportunidades y análisis causa-efecto
  - C. Problemas económicos, oportunidades y análisis causa-efecto
  - D. Problemas de control, oportunidades y análisis causa-efecto
  - E. Problemas de eficiencia, oportunidades y análisis causa-efecto
  - F. Problemas de servicio, oportunidades y análisis causa-efecto
- V. Recomendaciones detalladas (aproximadamente 5-10 páginas)
  - A. Objetivos y prioridades de mejora de sistemas
  - B. Restricciones
  - C. Plan de proyecto
    1. Reevaluación y refinamiento del alcance
    2. Plan maestro revisado
    3. Plan detallado para la fase de definición
- VI. Apéndices
  - A. Cualquier modelo de sistema detallado
  - B. Otros documentos según resulte apropiado

Esto concluye la fase del análisis del problema. Una de las siguientes decisiones debe realizarse luego de la conclusión de esta fase:

- Autorizar que el proyecto continúe, tal como está, a la fase de análisis de requerimientos.
- Ajustar el alcance, costo o programa para el proyecto y luego continuar con la fase de análisis de requerimientos.
- Cancelar el proyecto debido a: 1) falta de recursos para continuar el desarrollo del sistema, 2) darse cuenta de que los problemas y oportunidades simplemente no eran tan importantes como se había anticipado, o 3) comprensión de que no es probable que los beneficios del nuevo sistema excedan los costos.

Con cierto nivel de aprobación por parte de los PROPIETARIOS DEL SISTEMA, el proyecto puede ahora continuar hacia la fase de análisis de requerimientos.

## Fase de análisis de requerimientos

Muchos analistas inexpertos cometen un error crítico luego de completar la fase de análisis del problema. La tentación en ese punto es comenzar a considerar alternativas de solución, particularmente técnicas. Uno de los errores citados con mayor frecuencia en los nuevos sistemas de información se ilustra en la frase “seguro, el sistema funciona y técnicamente es impresionante, pero no hace lo que nosotros necesitamos”. La *fase de análisis de requerimientos* define los requerimientos de negocios para un sistema nuevo.

¿Notó la palabra clave en la oración citada? es “que” ¡y no “cómo”! Los analistas con frecuencia están tan preocupados por la solución *técnica* que inadecuadamente definen los requerimientos de *negocios* para esa solución. La fase de análisis de requerimientos responde la pregunta, “*¿Qué* necesitan y desean los usuarios de un sistema nuevo?

La fase de análisis de requerimientos es crítica para el éxito de cualquier sistema nuevo de información. En distintas metodologías la fase de análisis de requerimientos podría ser llamada *fase de definición* o *fase de diseño lógico*.

¿Se puede alguna vez pasar por alto la fase de análisis de requerimientos? ¡Definitivamente no! Los nuevos sistemas siempre serán evaluados, primero y sobre todo, acerca de si cumplen o no con los objetivos y requerimientos del negocio, ¡sin importar qué tan impresionante o compleja pueda ser la solución técnica!

Debe reconocerse que algunas metodologías integran las fases de análisis del problema y análisis de requerimientos en una sola fase.

Una vez más, sus componentes de sistemas de información (figura 4.14) pueden servir como un marco de referencia útil para documentar los requerimientos de sistemas de información. Nótese que seguimos ocupados con las perspectivas de los **USUARIOS DEL SISTEMA**. Los requerimientos pueden ser definidos en términos del marco laboral PIECES o en términos de los tipos de datos, procesos e interfaces que deben ser incluidos en el mismo.

En la figura 4.15 se ilustran las tareas típicas de la fase de análisis de requerimientos. El producto y la meta a alcanzar de la fase final permitirán crear una **DEFINICIÓN DE LOS REQUERIMIENTOS DEL NEGOCIO** que satisfaga los objetivos de mejora del sistema identificados en la fase anterior. Una de las primeras cosas que puede notar en este diagrama de tareas es que la mayoría de las tareas no se dan en forma secuencial como las de los diagramas de tareas anteriores. En lugar de eso, muchas de estas tareas ocurren en paralelo mientras el equipo trabaja hacia la meta de completar la definición de requerimientos. La fase de análisis de requerimientos generalmente incluye las siguientes tareas:

- 3.1 Identificar y expresar los requerimientos del sistema.
- 3.2 Priorizar los requerimientos de sistema.
- 3.3 Actualizar o refinar el plan de proyecto.
- 3.4 Comunicar la definición de requerimientos.

Ahora analicemos cada una de estas tareas con mayor detalle.

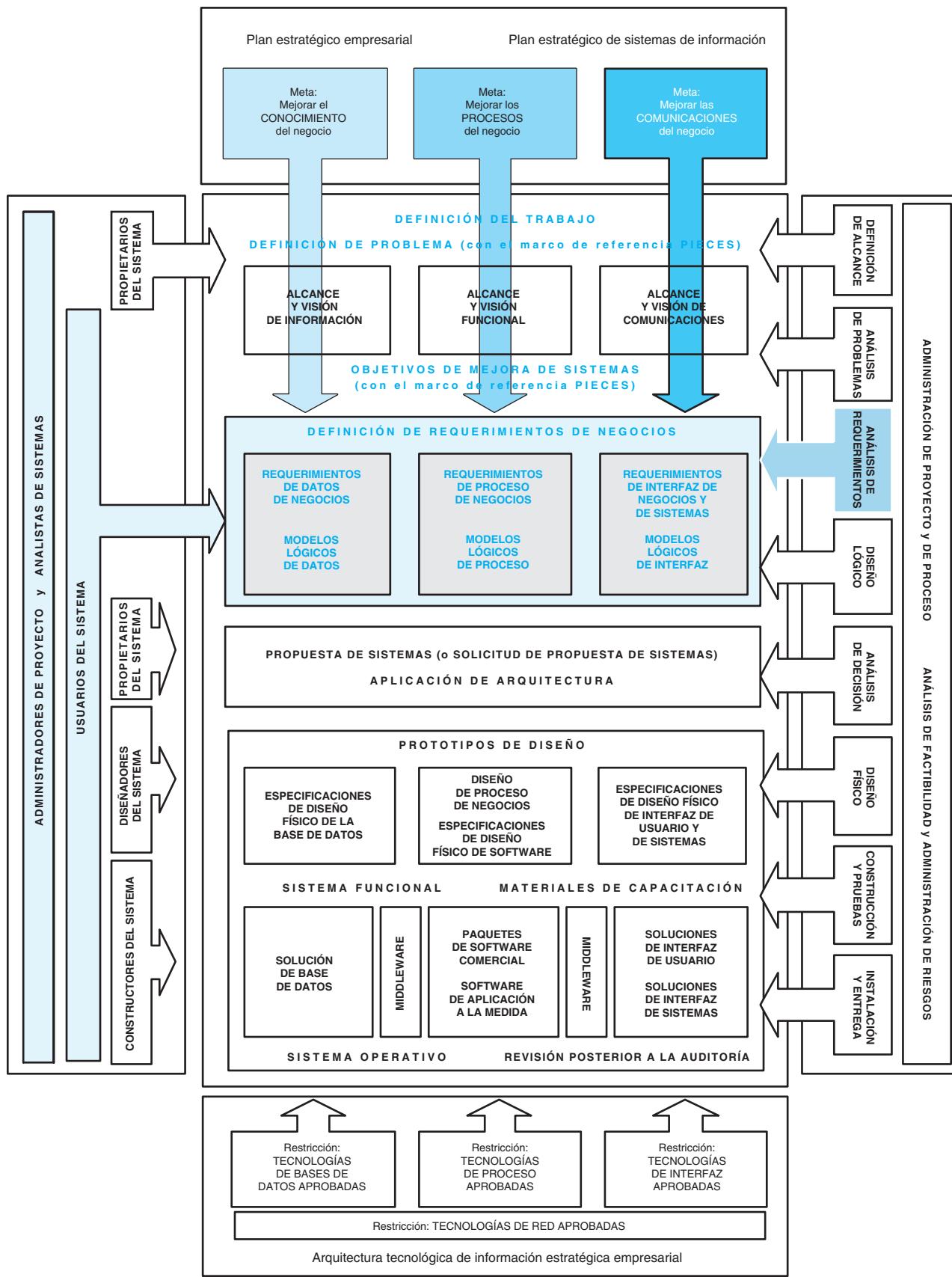
### > Tarea 3.1: Identificar y expresar los requerimientos del sistema

La tarea inicial de la fase de análisis de requerimientos es *identificar y expresar requerimientos*. Mientras que esto puede parecer como una tarea fácil o trivial, a menudo es la fuente de muchos errores, omisiones y conflictos. La base para esta tarea se estableció en la fase de análisis del problema cuando identificamos objetivos de mejora de sistemas. En forma mínima, esta tarea traduce estos objetivos en un esquema de **requerimientos funcionales** y **no funcionales** que se necesitarán para cumplir con los objetivos. Los requerimientos funcionales son frecuentemente identificados en términos de entradas, salidas, procesos y datos almacenados que son necesarios para satisfacer los objetivos de mejora del sistema. Ejemplos de requerimientos no funcionales incluyen desempeño (tiempo de desempeño y de respuesta); facilidad de aprendizaje y uso; presupuestos, costos y ahorros de costos; cronogramas y vencimientos; documentación y necesidades de capacitación; administración de la calidad y controles de auditoría interna y seguridad.

Rara vez esta tarea de definición identifica *todos* los requerimientos de negocios funcionales y no funcionales. Pero este esquema enmarcará su pensamiento mientras procede con tareas posteriores que agregarán nuevos requerimientos y detalles al mismo. Por tanto, ni la totalidad ni la perfección son una meta en esta tarea.

**requerimiento funcional**  
Descripción de las actividades y servicios que debe brindar un sistema.

**requerimiento no funcional** Descripción de otras características y restricciones que definen un sistema satisfactorio.



**FIGURA 4.14** Contexto de la fase de análisis de requerimientos del análisis de sistemas

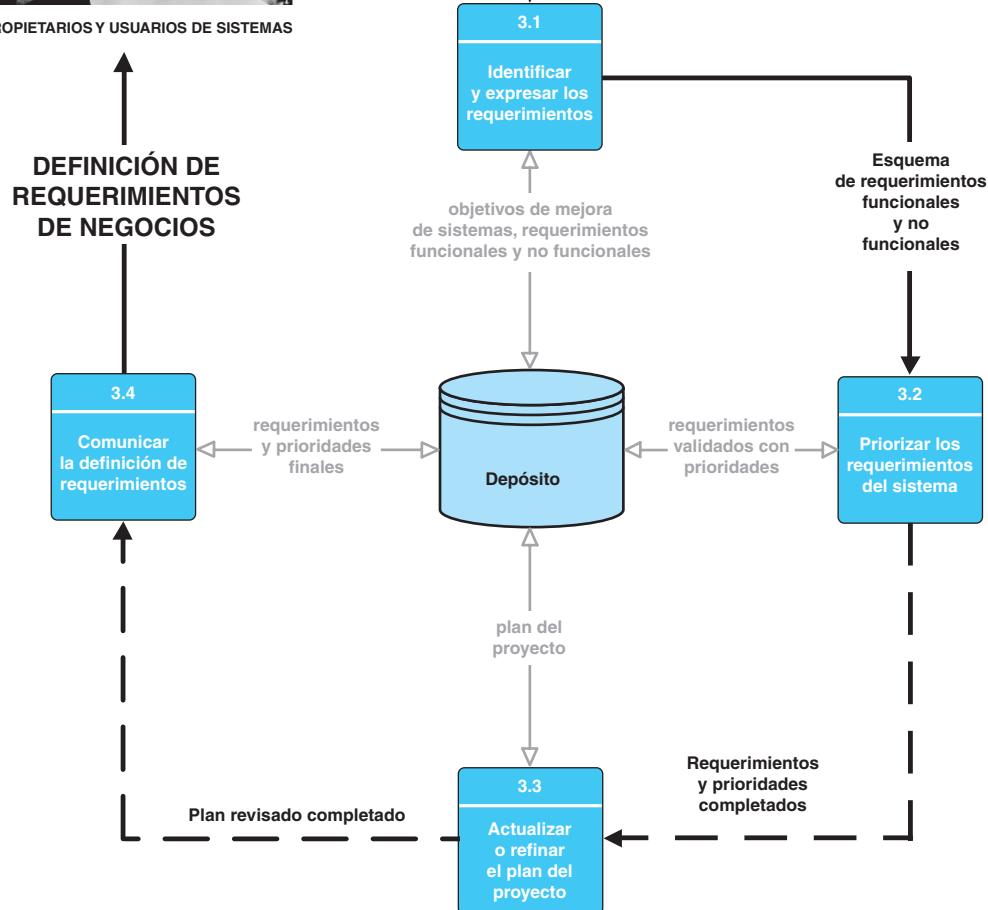
## LA COMUNIDAD DE NEGOCIOS



PROPIETARIOS Y USUARIOS DE SISTEMAS

**FIGURA 4.15**

Tareas para la fase de análisis de requerimientos del análisis de sistemas



Los ANALISTAS DEL SISTEMA facilitan la tarea. También documentan los resultados. Evidentemente, los USUARIOS DEL SISTEMA son la fuente principal de los requerimientos de negocios. Algunos PROPIETARIOS DEL SISTEMA pueden elegir participar en esta tarea ya que tuvieron un papel en la estructuración de los objetivos de mejora del sistema que guiarán la tarea. Los DISEÑADORES Y CONSTRUCTORES DE SISTEMAS no deben participar porque tienden a redirigir prematuramente el enfoque a la tecnología y a las soluciones técnicas.

Como se muestra en la figura 4.15, esta tarea (y fase) se dispara cuando se tiene la APROBACIÓN PARA CONTINUAR EL PROYECTO durante la FASE DE ANÁLISIS DEL PROBLEMA. La entrada fundamental son los OBJETIVOS DE MEJORA DEL SISTEMA definidos desde la fase de análisis del problema (vía repositorio). Desde luego, toda la información relevante obtenida durante la fase de análisis de problemas está disponible para su referencia según sea requerido.

El único producto a obtener en esta tarea son los REQUERIMIENTOS FUNCIONALES Y NO FUNCIONALES. Diversos formatos pueden funcionar. En su formato más simple, el esquema puede ser dividido en cuatro secciones lógicas: la lista original de los objetivos de mejora de sistemas, una sublista de *a)* entradas, *b)* procesos, *c)* salidas y *d)* datos almacenados necesarios para satisfacer el objetivo. Sin embargo, cada vez más, los analistas de sistemas expresan requerimientos funcionales mediante una herramienta de elaboración de modelos llamada **casos de uso**. Los casos de uso elaboran modelos de escenarios de negocios

---

**caso de uso** Escenario de negocios o evento respecto del cual el sistema debe proporcionar una respuesta definida. Los casos de uso evolucionaron a partir del análisis orientado a objetos; pero su utilización se ha vuelto común en muchos otros métodos de análisis y diseño de sistemas.

y sucesos que deben ser manejados por un sistema nuevo. Se presentan en el capítulo 6 y se utilizan a lo largo de este texto.

El marco de referencia PIECES que fue utilizado anteriormente para identificar problemas, oportunidades y restricciones también puede utilizarse como marco laboral para definir los requerimientos del borrador.

Diversas técnicas son aplicables a esta tarea. La planeación de requerimientos conjuntos (joint requirements planning, JRP) es la técnica preferida para delinejar con rapidez los requerimientos del negocio. De manera alternativa, los analistas podrían utilizar otros métodos de identificación de hechos como encuestas y entrevistas. Tanto el JRP como la identificación de hechos se enseñan en el siguiente capítulo.

### > Tarea 3.2: Priorizar los requerimientos del sistema

Anteriormente comentamos que el éxito de un proyecto de desarrollo de sistemas puede ser medido en términos del *grado* en el que se satisfacen los requerimientos del negocio. Pero no todos los requerimientos se crean igual. Si un proyecto se retrasa en horario o sobre el presupuesto, puede ser útil para reconocer qué requerimientos son más importantes que otros. Por tanto, dados los requerimientos validados, los propietarios del sistema y los usuarios del sistema deben *priorizar los requerimientos del sistema*.

Otorgar prioridades a los requerimientos puede facilitarse por medio de una técnica popular llamada *timeboxing*. El *timeboxing* intenta dividir requerimientos en “partes” que puedan ser implementadas dentro de un periodo que no acabe con la paciencia del usuario y de la comunidad administrativa. El *timeboxing* obliga a que las prioridades sean definidas claramente.

LOS ANALISTAS DE SISTEMAS facilitan la tarea de clasificación de prioridades. LOS PROPIETARIOS Y USUARIOS DEL SISTEMA establecen las prioridades actuales. LOS DISEÑADORES Y LOS CONSTRUCTORES DE SISTEMAS no participan en la tarea. La tarea es realizada cuando los REQUERIMIENTOS SON VALIDADOS. Debe resultar evidente que usted no pueda otorgar prioridades adecuadamente a un conjunto de requerimientos incompleto. El producto de esta tarea es obtener REQUERIMIENTOS CON PRIORIDADES. Las prioridades pueden ser clasificadas de acuerdo con su importancia relativa:

- Un *requerimiento obligatorio* es aquél que debe ser satisfecho por la mínima versión del sistema (versión 1.0.) El sistema es inútil sin él. ¡Tenga cuidado!, existe la tentación de etiquetar demasiados requerimientos como obligatorios. Un requerimiento obligatorio no puede ser clasificado porque es esencial para cualquier solución. De hecho, si un requerimiento obligatorio puede ser clasificado en orden de importancia, en realidad es un requerimiento deseable.
- Un *requerimiento deseable* es aquél que no es absolutamente esencial a la versión mínima del sistema (versión 1.0). Puede aún ser esencial para la visión de alguna versión futura. Los requerimientos deseables pueden y deben ser clasificados. El uso de números de versión como esquema de clasificación es una forma eficaz de comunicar y categorizar los requerimientos deseables.

### > Tarea 3.3: Actualizar o refinar el plan del proyecto

Nuevamente, recordemos que el alcance del proyecto es un objetivo móvil. Ahora que hemos identificado los requerimientos del sistema, debemos regresar y redefinir nuestra comprensión del alcance del proyecto y actualizar en consecuencia nuestro plan de proyecto. El equipo debe considerar la posibilidad de que el nuevo sistema pueda ser más grande de lo originalmente esperado. Si es así, en consecuencia, el equipo debe ajustar el programa, presupuesto o alcance. Debemos también asegurar la aprobación para que el proyecto continúe hacia la siguiente fase. (Puede que el trabajo ya se haya iniciado en las fases de diseño; sin embargo, las decisiones todavía requieren una revisión.)

El administrador del proyecto, en conjunto con los PROPIETARIOS DEL SISTEMA y el equipo de proyecto completo, facilitan esta tarea. Como siempre, el administrador del proyecto y los PROPIETARIOS DEL SISTEMA son los individuos clave en esta tarea. Deben considerar la posibilidad de que los requerimientos ahora excedan la visión original que se estableció para el proyecto y el nuevo sistema. Pueden tener que reducir el alcance para cumplir con un vencimiento o incrementar el presupuesto para que el trabajo se realice.

Como se muestra en la figura 4.15, esta tarea se realiza cuando los REQUERIMIENTOS SON PRIORIZADOS Y COMPLETADOS. El PLAN DEL PROYECTO actualizado es la otra entrada clave y se actualiza en el repositorio cuando se considere apropiado.

**timeboxing** Técnica que entrega funcionalidad y requerimientos de sistemas de información mediante el control de versiones. El equipo de desarrollo selecciona el subconjunto más pequeño del sistema que al ser puesto en práctica por completo genera valor inmediato para los propietarios y usuarios del sistema. Se desarrolla ese subconjunto, de preferencia en seis a nueve meses o menos. En forma subsiguiente, se desarrollan versiones del sistema con valor añadido, en marcos cronológicos similares.

## > Tarea 3.4: Comunicar la definición de requerimientos

La comunicación es una tarea continua de la fase de análisis de requerimientos. A lo largo de la fase, debemos comunicar requerimientos y prioridades para la comunidad de negocios. Los usuarios y administradores con frecuencia abogarán por una consideración de requerimientos y prioridades. La comunicación es el proceso a través del cual se deben mediar las diferencias de opinión. El administrador del proyecto y el patrocinador ejecutivo conjuntamente deben facilitar esta tarea. Actualmente, una intranet o un portal de proyecto se utilizan con frecuencia para comunicar los requerimientos. Algunos sistemas permiten a los usuarios y administradores *acceder* a los documentos de requerimientos para asegurarse de ser notificados conforme ocurran los cambios. Las habilidades interpersonales, de comunicación y negociación son esenciales para esta tarea.

### > Manejo de requerimientos permanentes

La fase de análisis de requerimientos está ahora completa. ¿O no? Alguna vez fue popular congelar los requerimientos del negocio antes de empezar las fases de diseño de sistemas y las de construcción. Pero la economía actual se ha vuelto cada vez más acelerada. Los negocios se miden de acuerdo con su capacidad para adaptarse con rapidez a requerimientos y oportunidades constantemente cambiantes. Los sistemas de información no pueden ser menos cambiantes que el negocio mismo. Por tanto, el análisis de requerimientos realmente nunca termina. Mientras que hacemos una transición silenciosa hacia las fases restantes de nuestro proyecto, continúa una necesidad constante de manejar requerimientos a través del curso del proyecto y la vida del sistema.

El manejo de los requerimientos define un proceso para los propietarios, usuarios, analistas, diseñadores y constructores del sistema para enviar los cambios propuestos a los requerimientos de un sistema. El proceso especificó cómo se deben solicitar y documentar los cambios, cómo se registrarán y rastrearán y cuándo y cómo se evaluarán para saber su prioridad y la forma en que eventualmente serán satisfechos (si alguna vez sucede).

## Fase de diseño lógico

No todos los proyectos aceptan el desarrollo basado en modelos, pero la mayoría incluye un poco de elaboración de modelos de sistemas. Un diseño lógico documenta más aún los requerimientos de negocios por medio de los modelos de sistemas que ilustran las estructuras de datos, procesos de negocios, flujos de datos e interfaces de usuarios (cada vez más por medio de modelos de objetos, como se presentó anteriormente en el capítulo). En un sentido, validan los requerimientos establecidos en la fase anterior.

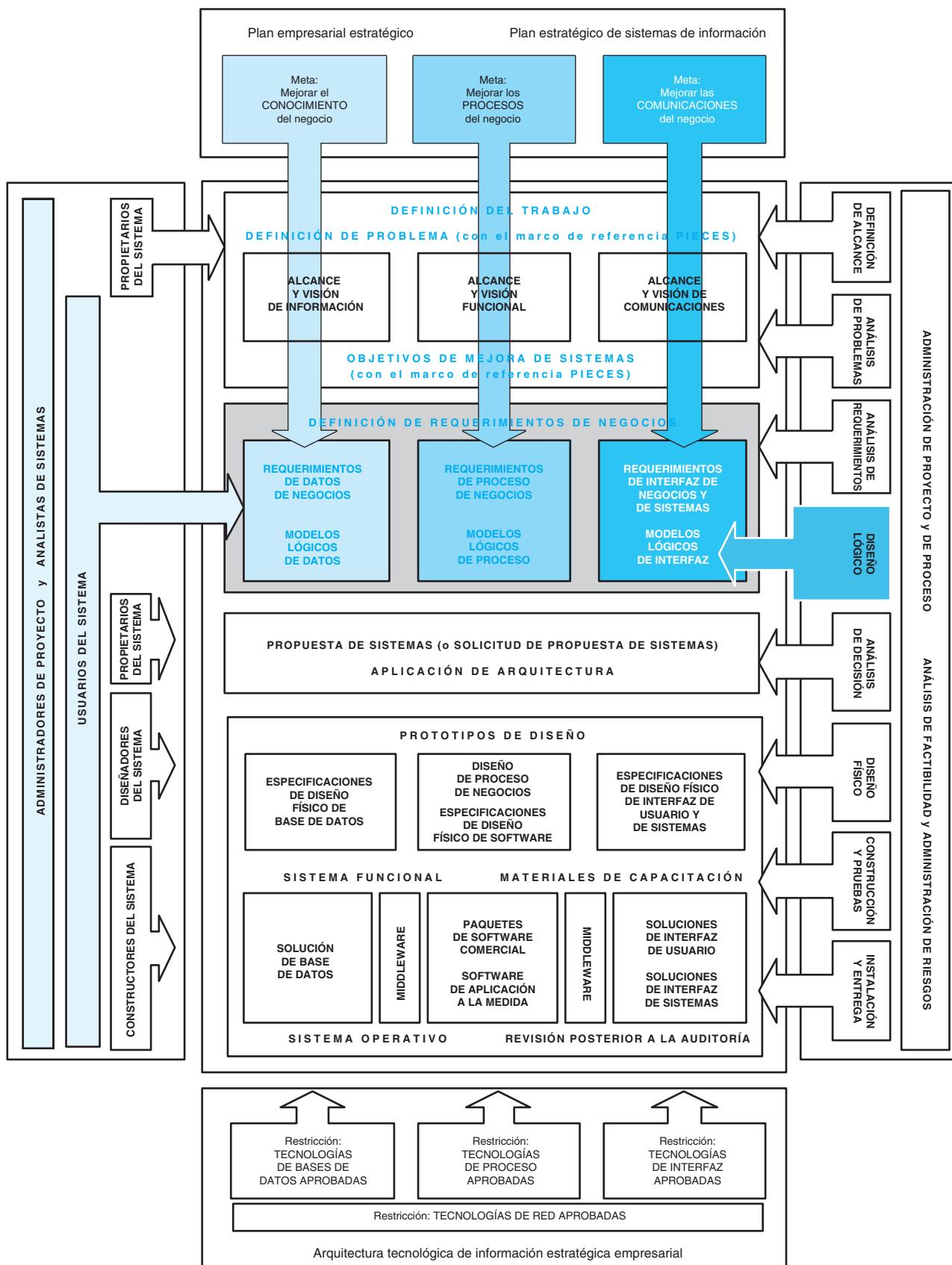
Una vez más, los componentes de sus sistemas de información (figura 4.16) pueden servir como un marco útil para documentar los requerimientos de sistemas de información. Nótese que todavía estamos ocupados por las perspectivas de los USUARIOS DEL SISTEMA. En esta fase, elaboramos diversos modelos de sistemas para documentar los requerimientos para un sistema nuevo y mejorado. Los modelos describen diversos aspectos de nuestros componentes. De manera alternativa, los prototipos podrían ser construidos para “identificar requerimientos”. Los prototipos de identificación fueron presentados anteriormente en el capítulo. Recuerde que a algunos prototipos se les puede aplicar una ingeniería inversa para convertirlos en modelos de sistemas.

En la figura 4.17 se ilustran las tareas típicas de la fase de diseño lógico. El producto y meta de la fase final es producir una DEFINICIÓN DE REQUERIMIENTOS DEL NEGOCIO que satisfagan los objetivos de mejora de sistemas identificados en la fase previa. Una de las primeras cosas que usted notará en este diagrama de tareas es que la mayoría de las tareas no son secuencias como en los diagramas de tareas anteriores. En lugar de eso, muchas de estas tareas ocurren en paralelo mientras el equipo trabaja hacia la meta de completar la definición de requerimientos.

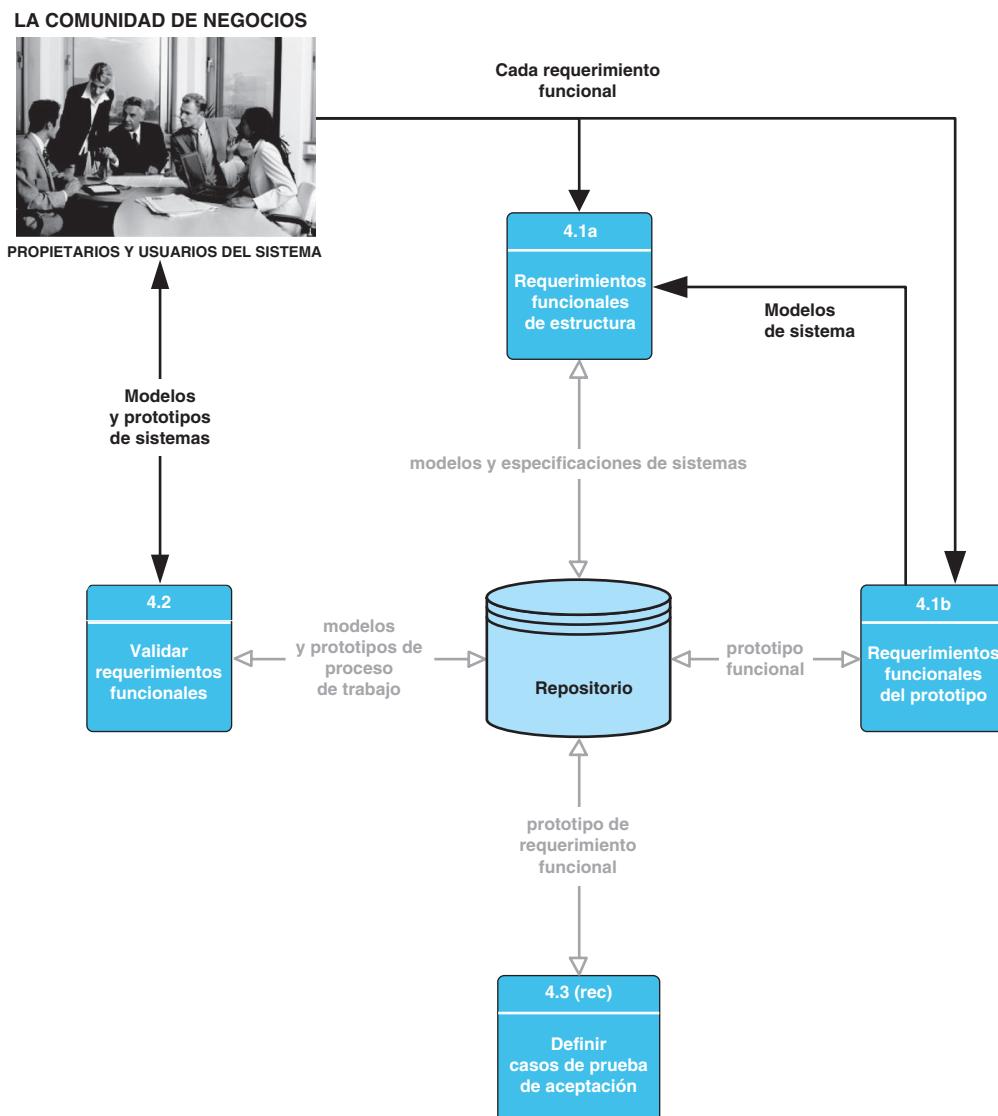
La fase de diseño lógico generalmente incluye las siguientes tareas:

- 4.1a Requerimientos funcionales de estructura
- 4.1b Requerimientos funcionales del prototipo
- 4.2 Validar requerimientos funcionales
- 4.3 Definir casos de prueba de aceptación

Ahora analicemos cada una de estas tareas con mayor detalle.



**FIGURA 4.16** Contexto de la fase de diseño lógico del análisis de sistemas

**FIGURA 4.17**

Tareas para la fase de diseño lógico del análisis de sistemas

### > Tarea 4.1a: Requerimientos funcionales de estructura

Un método para el diseño lógico es definir los *requerimientos funcionales de estructura*. Esto significa que, por medio de métodos acelerados, usted debe elaborar o actualizar uno o más modelos de sistemas para ilustrar el requerimiento funcional. Éstos pueden incluir cualquier combinación de datos, procesos y modelos de objetos que describan con precisión el negocio y los requerimientos de los usuarios (pero no una solución técnica). Los modelos de sistemas no están completos hasta que todos los requerimientos funcionales proporcionados han sido modelados. Los modelos con frecuencia se complementan con especificaciones lógicas detalladas que describen los atributos de datos, reglas y políticas de negocios.

Los analistas de sistemas facilitan la tarea. Ellos también documentan los resultados. Evidentemente, los **USUARIOS DE SISTEMAS** son la fuente primaria de detalles de hechos necesarios para elaborar los modelos. Como se muestra en la figura 4.17, esta tarea (y fase) es realizada por *cada REQUERIMIENTO FUNCIONAL*. Las salidas son los **MODELOS DEL SISTEMAS Y ESPECIFICACIONES DETALLADAS** reales. El nivel de detalle requerido depende de la metodología que se sigue. Los métodos acelerados generalmente requieren de documentación “únicamente esencial”. ¿Cuánto es esencial? Eso es discutible, pero los expertos hábiles en métodos sostienen que cada producto será esencial para las siguientes fases de diseño y programación. En este texto se le enseñará una diversidad de herramientas y técnicas de elaboración de modelos de sistemas para aplicarse a un diseño lógico.

### > Tarea 4.1b: Requerimientos funcionales del prototipo (alternativa)

La elaboración de prototipos es una alternativa (y a veces un requerimiento previo) para la elaboración de sistemas. Algunas veces los usuarios tienen dificultad para expresar los hechos necesarios para elaborar los modelos de sistemas adecuados. En tal caso, una alternativa o un enfoque complementario a la elaboración de modelos de sistemas es construir prototipos de identificación. La elaboración de prototipos normalmente se utiliza en la fase de análisis de requerimientos para construir entradas y salidas de ejemplos. Estas entradas y salidas ayudan a construir la base de datos y los programas para introducir y generar datos hacia y desde la base de datos. Aunque la elaboración del prototipo es opcional, frecuentemente se aplica a los proyectos de desarrollo de sistemas, en especial en casos donde los usuarios tengan dificultad en comunicar o visualizar sus requerimientos de negocio. La filosofía del prototipo afirma que los usuarios reconocerán sus requerimientos cuando los vean.

Los CONSTRUCTORES DE SISTEMAS facilitan esta tarea de análisis. Los ANALISTAS DE SISTEMAS documentan y analizan los resultados. Como siempre, los USUARIOS DE SISTEMAS son la fuente primaria de la entrada de hechos a la tarea. En la figura 4.17 se demuestra que esta tarea depende de uno o más REQUERIMIENTOS FUNCIONALES que han sido identificados por los usuarios. Los constructores y los analistas de sistemas responden construyendo PROTOTIPOS. Como se describió anteriormente en este capítulo, es posible aplicar una *ingeniería inversa* a algunos MODELOS DE SISTEMAS basándose en los prototipos de las bases de datos y las bibliotecas de programas.

### > Tarea 4.2: Validar requerimientos funcionales

Tanto los MODELOS DE SISTEMAS como los PROTOTIPOS son representaciones de los requerimientos de los usuarios. Éstos deben ser validados para que estén completos y correctos. Los ANALISTAS DE SISTEMAS facilitan la tarea de asignación de prioridades al comprometer en forma interactiva a los usuarios del sistema para la identificación de errores, omisiones y aclaraciones.

### > Tarea 4.3: Definir casos de prueba de aceptación

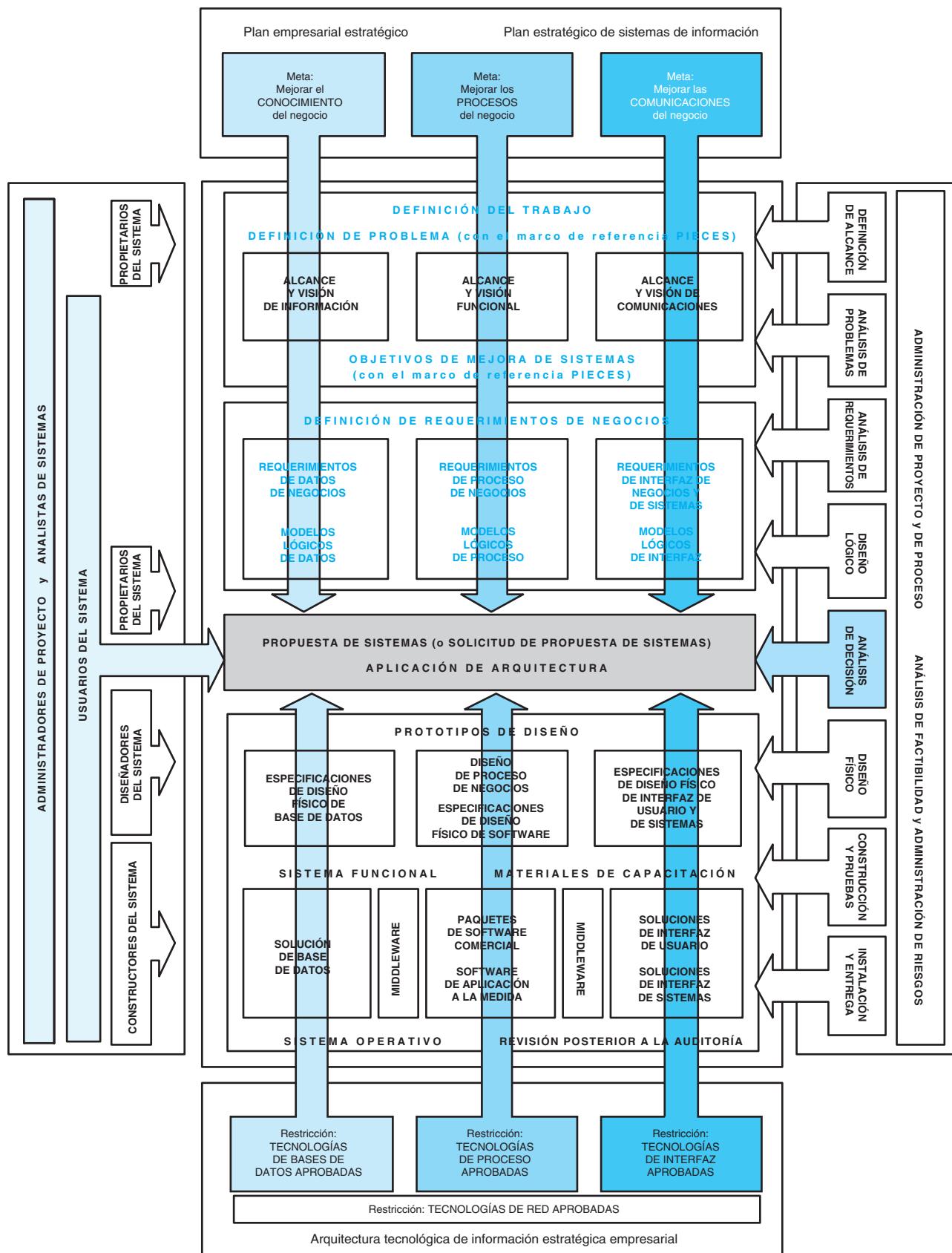
Aunque no es una tarea requerida, la mayoría de los expertos están de acuerdo en que no es demasiado adelantado comenzar a planear las pruebas del sistema. Los modelos y los prototipos de sistemas definen muy eficazmente los requerimientos de proceso, reglas de datos y reglas de negocios para el nuevo sistema. En consecuencia, estas especificaciones pueden ser utilizadas para definir CASOS DE PRUEBA que pueden finalmente ser utilizados para probar que los programas estén correctos. Tanto los ANALISTAS como los CONSTRUCTORES DE SISTEMAS pueden desempeñar esta tarea y validar los casos de pruebas con los USUARIOS DE SISTEMAS.

Recuerde que los OBJETIVOS DE MEJORA DE SISTEMAS fueron definidos anteriormente en el proyecto. Los casos de pruebas pueden ser definidos para probar estos objetivos también.

## Fase de análisis de decisión

Una vez que se tienen los requerimientos del negocio para un sistema de información mejorado, es posible abordar la implementación tecnológica del nuevo sistema (incluidas las alternativas basadas en computadora). El propósito de la *fase de análisis de decisión* es identificar las soluciones alternativas, analizarlas y recomendar un sistema objetivo que será diseñado, construido e implementado. Es muy posible que alguien ya haya detectado una posible solución técnica. Pero las soluciones alternativas, lo que es mejor, casi siempre existen. Durante la fase de análisis de decisión, es necesario que usted identifique opciones, las analice y luego venda la mejor solución con base en el análisis.

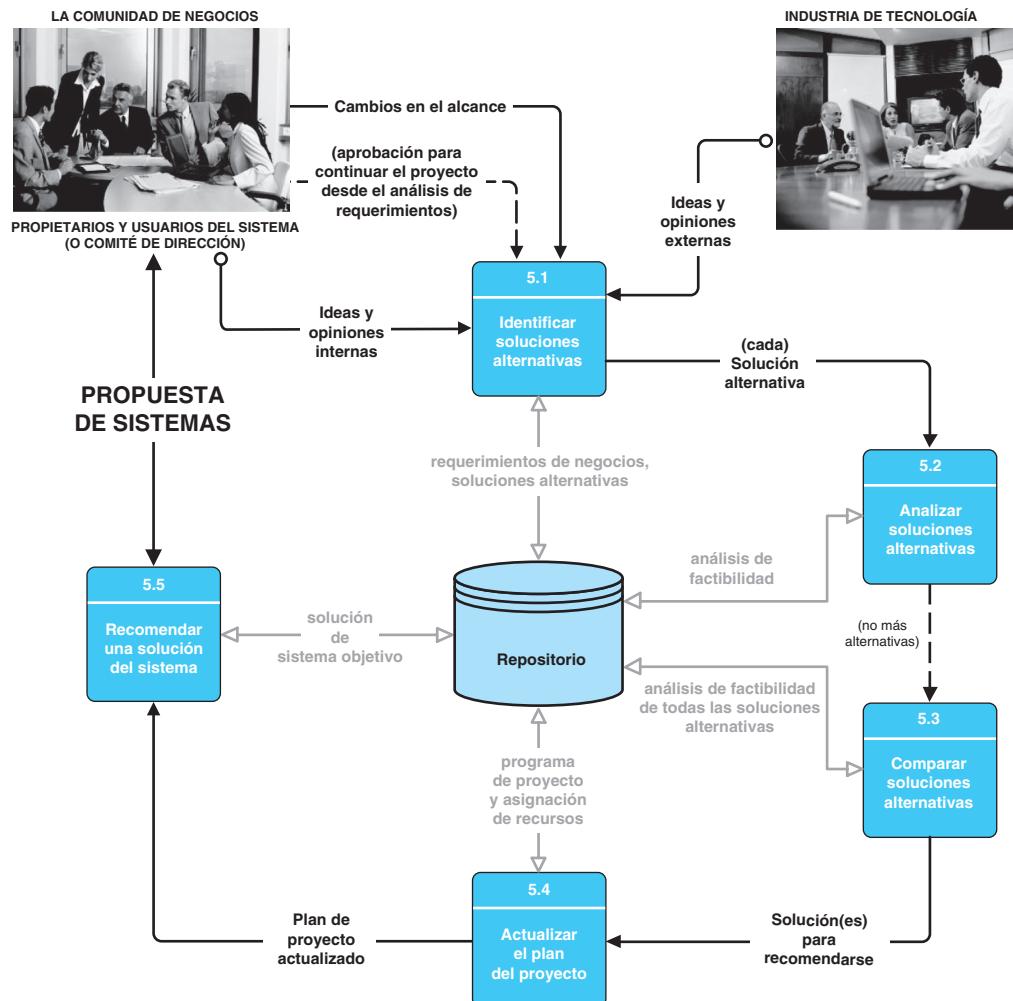
Una vez más, los componentes de sistemas de información (figura 4.18) pueden servir como un marco de referencia útil para la fase de análisis de decisión. Una de las primeras cosas que usted debe notar es que la tecnología de información y la arquitectura comienzan a influir en las decisiones que debemos tomar. En algunos casos, debemos trabajar con estándares. En otros casos, podemos intentar aplicar tecnología distinta o



**FIGURA 4.18** Contexto de la fase de análisis de decisión del análisis de sistemas

**FIGURA 4.19**

Tareas para la fase de análisis de decisión del análisis de sistemas



en surgimiento. Usted también debe notar que las perspectivas (puntos de vista) de los USUARIOS DEL SISTEMA están en transición con respecto a las de los DISEÑADORES DEL SISTEMA. Nuevamente, esto refleja nuestro interés por los negocios o la tecnología. Pero aún así, no estamos en el diseño. Los componentes indican nuestra meta de desarrollar una propuesta que cumpla con todos los requerimientos.

En la figura 4.19 se ilustran las tareas típicas de la fase de análisis de decisión. El producto y meta de la fase final es producir una PROPUESTA DEL SISTEMA que satisfaga los requerimientos de sistemas identificados en la fase previa. La fase de análisis de decisión normalmente incluye las siguientes tareas:

- 5.1 Identificar soluciones alternativas
- 5.2 Analizar soluciones alternativas
- 5.3 Comparar soluciones alternativas
- 5.4 Actualizar el plan del proyecto
- 5.5 Recomendar una solución del sistema

Ahora analicemos con mayor detalle cada una de estas tareas.

#### > Tarea 5.1: Identificar soluciones alternativas

Dados los requerimientos de negocios establecidos en la fase de definición del análisis de sistemas, debemos primero identificar las soluciones alternativas. Algunas soluciones alternativas serán planteadas por las ideas y opiniones de diseño de los PROPIETARIOS Y USUARIOS DE SISTEMAS. Otras vendrán de diversas fuentes incluyendo a los ANALISTAS DE SISTEMAS, DISE-

ÑADORES DE SISTEMAS, consultores técnicos y otros profesionales de sistemas de información. Y algunas opciones pueden estar limitadas por una arquitectura de tecnología aprobada, predefinida. Es la intención de esta tarea no evaluar las opciones, sino simplemente definir las posibles alternativas de solución que se van a considerar.

LOS ANALISTAS DE SISTEMAS facilitan esta tarea. LOS PROPIETARIOS Y USUARIOS DE SISTEMAS normalmente no participan de manera directa en esta tarea, sino que pueden contribuir con ideas y opiniones que comienzan la tarea. Por ejemplo, un propietario o usuario puede haber leído un artículo o haber escuchado o sabido de la forma en que se implementó un sistema similar de un conocido o un competidor. En cualquier caso, es políticamente profundo el considerar las ideas. Los DISEÑADORES Y CONSTRUCTORES DE SISTEMAS como administradores de bases de datos, los administradores de redes y los arquitectos de tecnología y los programadores también son una fuente de ideas y opiniones.

Como se mostró en la figura 4.19, esta tarea es formalmente realizada cuando se otorga la APROBACIÓN PARA CONTINUAR EL PROYECTO DE LA FASE DE ANÁLISIS DE REQUERIMIENTOS. En realidad, las ideas y opiniones han sido generadas y capturadas desde la fase de investigación preliminar; es humano sugerir soluciones a lo largo de cualquier proceso de solución de problemas. Nótese que, además de venir del equipo de proyecto mismo, las IDEAS Y OPINIONES pueden ser generadas de fuentes internas y externas. Cada idea generada se considera como una SOLUCIÓN ALTERNATIVA para los REQUERIMIENTOS DEL NEGOCIO.

La cantidad de información que describe las características de cualquier solución opcional puede resultar abrumadora. Una matriz de alternativas, como la figura 4.20, es una herramienta útil para capturar, organizar y comparar eficazmente las características de las distintas opciones de solución.

Como ha sido el caso a lo largo de este capítulo, las técnicas de identificación de hechos y de facilitación de grupo como JRP son las técnicas principales utilizadas para investigar soluciones alternativas de sistemas. Las técnicas de identificación de hechos y de facilitación de grupo se enseñan en el siguiente capítulo. Asimismo, en el capítulo 9, “Análisis de factibilidad y propuesta del sistema”, se le enseñará la forma de generar y documentar las soluciones alternativas de sistemas en la matriz.

## > Tarea 5.2: Analizar soluciones alternativas

Debe analizarse la factibilidad de cada solución alternativa de sistemas. Esto puede ocurrir conforme se identifique cada posible solución o después de que todas las soluciones opcionales han sido identificadas. Los análisis de factibilidad no deben ser limitados a los costos y beneficios. La mayoría de los analistas evalúan soluciones contra al menos cuatro conjuntos de criterios:

- *Factibilidad técnica.* ¿Es la solución técnicamente práctica? ¿Nuestro personal tiene experiencia técnica para diseñar y construir esta solución?
- *Factibilidad operativa.* ¿La solución cumplirá con todos los requerimientos de los usuarios? ¿A qué grado? ¿Cómo cambiará la solución el ambiente de trabajo del usuario? ¿Cómo se sienten los usuarios acerca de dicha solución?
- *Factibilidad económica.* ¿La solución tiene un costo-beneficio?
- *Factibilidad de programa.* ¿Puede la solución ser diseñada e implementada dentro de un periodo aceptable?

Al completar esta tarea, los analistas y usuarios deben tener cuidado de no hacer comparaciones entre las diferentes soluciones. El análisis de factibilidad se realiza con cada solución alternativa sin importar la factibilidad de las demás soluciones. Este enfoque desalienta al analista y a los usuarios de tomar una decisión en forma prematura con respecto a una solución.

De nuevo, los ANALISTAS DE SISTEMAS facilitan la tarea. Normalmente los PROPIETARIOS DEL SISTEMA y los USUARIOS DEL SISTEMA analizan la factibilidad operativa, económica y de programa. Los DISEÑADORES Y CONSTRUCTORES DEL SISTEMA normalmente contribuyen al análisis y juegan un papel fundamental para analizar la factibilidad técnica.

En la figura 4.19 se muestra que la tarea es realizada cuando cada solución alternativa es analizada; sin embargo, es aceptable retrasar esta tarea hasta que todas las soluciones hayan sido identificadas. Las opiniones sobre los análisis de factibilidad provienen de diversos participantes en el equipo; sin embargo, es común que los expertos externos (e

**FIGURA 4.20** Matriz de soluciones alternativas del sistema

Características	Solución alternativa 1	Solución alternativa 2	Solución alternativa 3	Solución alternativa...
<b>Porción de sistema computarizado</b> Breve descripción de esa porción del sistema que sería computarizada en esta solución alternativa.	Paquete COTS Platinum Plus de Entertainment Software Solutions se adquiriría y sería personalizada para satisfacer la funcionalidad requerida para satisfacer los servicios del negocio.	Servicios y operaciones del almacén en relación con el llenado de pedidos.	Igual a la solución alternativa 2.	
<b>Beneficios</b> Breve descripción de los beneficios de negocios que se lograrían con esta solución alternativa.	Esta solución puede ser implementada rápidamente porque es comprada.	Respalda totalmente los procesos de negocios requeridos para SoundStage Inc. Más interacción, más eficiencia con las cuentas del negocio.	Igual a la solución alternativa 2.	
<b>Servidores y estaciones de trabajo</b> Descripción de los servidores y las estaciones de trabajo necesarias para respaldar esta solución alternativa.	La arquitectura técnica sugiere utilizar una computadora con procesador Pentium Pro, servidores de clase con MS Windows NT y CPU Pentium, estaciones de trabajo con MS Windows NT 4.0 (clientes).	Igual a la solución alternativa 1.	Igual a la solución alternativa 1.	
<b>Herramientas de software necesarias</b> Las herramientas de software necesarias para diseñar y construir la solución alternativa (por ejemplo, sistema de administración de base de datos, emuladores, sistemas operativos, lenguajes, etcétera). No son generalmente aplicables si los paquetes de aplicación de software se van a adquirir.	MS Visual C++ y MS Access para proporcionar el reporte y la integración.	MS Visual Basic 5.0 System Architect 3.1 Internet Explorer	MS Visual Basic 5.0 System Architect 3.1 Internet Explorer	
<b>Software de aplicación</b> Descripción del software que se va a adquirir, desarrollar, accesar o alguna combinación de estas técnicas.	Solución del paquete	Solución personalizada	Igual a la solución alternativa 2.	
<b>Método de proceso de datos</b> Generalmente alguna combinación del procesamiento en línea, por lotes, lotes diferidos, lotes remotos y tiempo real.	Cliente/servidor	Igual a la solución alternativa 1.	Igual a la solución alternativa 1.	
<b>Dispositivos e implicaciones de salida</b> Descripción de dispositivos de salida que se utilizarán, requerimientos de salida especiales (por ejemplo, red, formatos preimpresos, etcétera) y consideraciones de salida (por ejemplo, restricciones de tiempo).	(2) HP4MV impresoras láser del departamento (2) impresoras láser HP5SI	(2) HP4MV impresoras láser por departamento. (2) HP5SI LAN impresoras láser. (1) PRINTRONIX impresora de código de barras (incluye programas y controladores). Las páginas Web deben ser diseñadas para resolución VGA. Todas las pantallas internas serán diseñadas para resolución SVGA.	Igual a la solución alternativa 2.	
<b>Dispositivos e implicaciones de entrada</b> Descripción de métodos de entrada que se van a utilizar, dispositivos de entrada (tablero, ratón, etcétera), requerimientos especiales de entrada (por ejemplo, formatos nuevos o revisados de los cuales se ingresarán datos) y consideraciones de entrada (por ejemplo, tiempos de las entradas reales).	Tablero y ratón	Apple "Quick Take" cámara digital y software. (1.5) PSC Quickscan escáneres de código de barras láser. (1) HP Scanjet 4C escáner flatbed. Teclado y ratón.	Igual a la solución alternativa 2.	
<b>Dispositivos e implicaciones de almacenamiento</b> Breves descripciones de qué datos deben ser almacenados, qué datos serían consultados de las tiendas existentes, qué medios de almacenamiento se utilizarían, cuánta capacidad de almacenamiento sería necesaria y cuántos datos serían organizados.	Servidor MS SQL DBMS con capacidad de 100 GB.	Solución alternativa 1	Igual a la solución alternativa 1.	

influencias) también proporcionen información al respecto. El análisis de factibilidad de cada solución alternativa se guarda en el repositorio para hacer una comparación posterior con las demás opciones de solución.

Nuevamente, las técnicas de identificación de hechos, desempeñan un papel importante en la tarea de análisis de sistemas. En consecuencia, la realización de un análisis de factibilidad para cada solución alternativa del sistema es esencial. Esa técnica se enseña en el capítulo 9, "Análisis de factibilidad y propuesta del sistema".

### > Tarea 5.3: Comparar soluciones alternativas

Una vez que el análisis de factibilidad ha sido completado para cada solución alternativa, podemos compararlas y elegir una o más soluciones para recomendarlas a los PROPIETARIOS y a los USUARIOS DE SISTEMAS. En este punto, cualquier solución no factible normalmente se elimina de cualquier consideración posterior. Debido a que estamos buscando la solución más factible de las soluciones restantes, identificaremos y recomendaremos la solución que ofrezca la más completa combinación de factibilidad técnica, operativa, económica y de cronograma. Debe entenderse que durante la selección, es raro que se encuentre una solución que presente la factibilidad más operativa, técnica, económica y de cronograma.

Una vez más, los ANALISTAS DEL SISTEMA facilitan la tarea. Los DISEÑADORES Y CONSTRUCTORES DEL SISTEMA deben estar disponibles para responder cualquier pregunta de factibilidad técnica. Pero finalmente, los PROPIETARIOS y los USUARIOS DEL SISTEMA deben estar facultados para dirigir el análisis final y la recomendación.

En la figura 4.19, esta tarea se lleva a cabo cuando se finaliza el análisis de factibilidad de todas las soluciones alternativas (NO MÁS SOLUCIONES ALTERNATIVAS). La información de entrada son TODOS LOS ANÁLISIS DE FACTIBILIDAD DE LAS SOLUCIONES ALTERNATIVAS. Una vez más, se puede utilizar una matriz para comunicar la gran cantidad de información que se tiene sobre las opciones de solución. La matriz de factibilidad de la figura 4.21 permite una comparación lado a lado de los diversos análisis de factibilidad que se tienen sobre las diferentes soluciones alternativas.

El producto de esta tarea es la(s) SOLUCIÓN(ES) QUE SE RECOMENDARÁ(N). Si se recomienda más de una solución, se deben establecer prioridades.

De nuevo, las técnicas de análisis de factibilidad (y la matriz) se enseñarán en el capítulo 9, "Análisis de factibilidad y propuesta de sistema".

### > Tarea 5.4: Actualizar el plan del proyecto

Esperamos que se haya percatado del tema recurrente a lo largo de este capítulo. Continuamente actualizamos nuestro plan de proyecto al tiempo que aprendemos más acerca de un sistema, sus problemas, sus requerimientos y sus soluciones. Ajustamos el alcance en consecuencia. Por tanto, con base en nuestras soluciones recomendadas, debemos una vez más reevaluar el alcance del proyecto y, en consecuencia, *actualizar el plan del proyecto*.

El administrador del proyecto, en conjunto con los PROPIETARIOS DE SISTEMAS y el equipo completo del proyecto, facilitan esta tarea. Los ANALISTAS DE SISTEMAS y los PROPIETARIOS DE SISTEMAS son los individuos fundamentales en esta tarea. Pero como estamos en transición hacia el diseño de sistema técnico, necesitamos comenzar a incluir a los DISEÑADORES Y CONSTRUCTORES DE SISTEMAS en las actualizaciones del plan del proyecto.

Como se muestra en la figura 4.19, esta tarea se dispara por la terminación de la(s) SOLUCIÓN(ES) QUE SE RECOMENDARÁ(N). El más reciente PROGRAMA DE PROYECTO Y ASIGNACIÓN DE RECURSOS debe ser revisado y actualizado. El PLAN DE PROYECTO ACTUALIZADO es el producto clave. El plan actualizado debe ahora incluir un plan detallado para la fase de diseño de sistemas que seguirá.\*

### > Tarea 5.5: Recomendar una solución del sistema

Al igual que con las fases de investigación preliminar y de análisis de problemas, la fase de análisis de decisión concluye con una tarea de comunicación. Debemos *recomendar una solución de sistema* para la comunidad del negocio.

\*Nota del R.T.: Para el lector interesado en el tema, las técnicas y pasos para actualizar el plan de proyecto se enseñan en el PMBOK del PMI.

**FIGURA 4.21** Matriz de soluciones alternativas del sistema

Criterio de factibilidad	Peso	Solución alternativa 1	Solución alternativa 2	Solución alternativa 3	Solución alternativa...
<b>Factibilidad operativa</b> <b>Funcionalidad.</b> Descripción de a qué grado la solución alternativa beneficiaría a la organización y qué tan bien trabajaría el sistema. <b>Política.</b> Descripción de qué tan bien recibida sería esta solución por parte de la administración de usuarios, y el resto de la organización.	30%	Respalda únicamente los requerimientos de servicios ya que los procesos de negocios actuales tendrían que ser modificados para aprovechar la funcionalidad del software.	Respalda totalmente la funcionalidad requerida del usuario.	Igual a la solución alternativa 2.	
		<b>Calificación: 60</b>	<b>Calificación: 100</b>	<b>Calificación: 100</b>	
<b>Factibilidad técnica</b> <b>Tecnología.</b> Evaluación de la madurez, disponibilidad (o capacidad de adquirir) y necesidad de la tecnología de cómputo requerida para respaldar esta solución alternativa.  <b>Experiencia.</b> Evaluación de la experiencia técnica necesaria para desarrollar, operar y mantener la solución alternativa del sistema.	30%	La liberación actual del paquete Platinum Plus es la versión 1.0 y ha estado en el mercado por sólo seis semanas. La madurez del producto es un riesgo y el proveedor carga una cuota mensual adicional al monto del software por incluir el soporte técnico del mismo.  Se requiere contratar o capacitar a un profesional que tenga experiencia en lenguaje C++ para realizar modificaciones para los requerimientos de integración del sistema.	Aunque el personal técnico actual tiene sólo experiencia en Powerbuilder, los analistas senior que vieron la demostración y presentación de MS Visual Basic han acordado que la transición será simple y que encontrar programadores experimentados en VB será más fácil que encontrar programadores de Powerbuilder y a un costo mucho menor.  MS Visual Basic 5.0 es una tecnología madura basada en el número de versión.	Aunque el personal técnico actual se siente cómodo con Powerbuilder, la administración está preocupada con la reciente adquisición de Powerbuilder de Sybase Inc. MS SQL Server es un estándar actual de la compañía y compite con SYBASE en el mercado de los DBMS cliente/servidor. Debido a esto no tenemos garantía de que las versiones futuras de Powerbuilder "correrán bien" con nuestra versión actual de SQL Server.	
		<b>Calificación: 50</b>	<b>Calificación: 95</b>	<b>Calificación: 60</b>	
<b>Factibilidad económica</b> <b>Costo para desarrollar:</b> <b>Período de retribución (descuento):</b> <b>Valor neto presente:</b> <b>Cálculos detallados:</b>	30%	Aproximadamente \$350 000  Aproximadamente 4.5 años Aproximadamente \$210 000 Véase el Anexo A	Aproximadamente \$418 040  Aproximadamente 3.5 años Aproximadamente \$306 748 Véase el Anexo A	Aproximadamente: \$400 000  Aproximadamente 3.3 años Aproximadamente \$325 500 Véase el Anexo A	
		<b>Calificación: 60</b>	<b>Calificación: 85</b>	<b>Calificación: 90</b>	
<b>Factibilidad del programa</b> Evaluación de cuánto tomará diseñar e implementar la solución alternativa	10%	Menos de tres meses.	9 a 12 meses	9 meses	
		<b>Calificación: 95</b>	<b>Calificación: 80</b>	<b>Calificación: 85</b>	
<b>Clasificación</b>	<b>100%</b>	<b>60.5</b>	<b>92</b>	<b>83.5</b>	

El administrador de proyecto y el patrocinador ejecutivo deben realizar conjuntamente esta tarea. Otros participantes en la reunión deben incluir al equipo completo del proyecto, así como a los PROPIETARIOS, USUARIOS, ANALISTAS, DISEÑADORES y CONSTRUCTORES DE SISTEMAS asignados. Como siempre, la reunión debe estar abierta a cualquier empleado interesado de la comunidad del negocio. También, si se estableció un sitio Web de intranet para el proyecto, debe ser actualizado a lo largo de las fases de análisis de problemas para asegurar una comunicación continua del progreso del proyecto.

Esta tarea se realiza cuando se tiene el PLAN DEL PROYECTO ACTUALIZADO. La SOLUCIÓN DE SISTEMA OBJETIVO (de la tarea 4.3) se reforma para su presentación como una PROPUESTA DE SISTEMA. El formato puede ser un informe, una presentación verbal o una inspección por un auditor o un grupo de colegas (llamado *walkthrough*). En la figura 4.22, se muestra un esquema de un informe escrito.

## La siguiente generación: Análisis de sistemas

Pronosticar el futuro del análisis de sistemas no es fácil, pero lo intentaremos. La tecnología CASE mejorará y eso facilitará el modelado de requerimientos de sistemas. Primero, las herramientas CASE incluirán el modelado de objetos para respaldar las nuevas técnicas de análisis orientadas a objetos. Mientras que algunas herramientas CASE serán sólo orientadas a objetos, creemos que la demanda de otros tipos de soporte de modelado (por ejemplo, modelado de datos para bases de datos, modelado de proceso para BPR) darán un atractivo adicional a las herramientas completas CASE que puedan soportar muchos tipos de modelos. Segundo, la tecnología de ingeniería inversa en las herramientas CASE mejorará más nuestra capacidad de generar con mayor rapidez el modelo inicial de sistemas a partir de las bases de datos y los programas de aplicación existentes.

Mientras tanto, la tecnología RAD continuará permitiendo los métodos de análisis acelerado tales como la elaboración de prototipos. También esperamos que la tendencia de que las herramientas RAD y CASE interactúen a través de la ingeniería hacia adelante e inversa simplifiquen aún más la elaboración de modelos y de prototipos de identificación.

El análisis orientado a objetos eventualmente reemplazará el análisis estructurado y la ingeniería de información como las mejores prácticas para el análisis de sistemas. Este cambio puede no ocurrir con tanta rapidez como los partidarios de los objetos desearían, pero ocurrirá bastante rápido para una generación de analistas de sistemas que sean hábiles en los métodos más antiguos. Existe una gran oportunidad para que

los analistas jóvenes y talentosos dirijan la transición; sin embargo, las oportunidades profesionales seguirán siendo fuertes para los analistas que sepan que la elaboración de modelos de datos continuará en uso para el diseño de bases de datos. También, el renacimiento de la elaboración de modelos de proceso continuará mientras que los proyectos BPR sigan proliferando.

Asimismo pronosticamos que nuestras propuestas de sistemas serán cada vez más interesantes. Mientras que el Internet, el comercio electrónico y los negocios electrónicos continúen su dominio en nuestra economía, los analistas de sistemas propondrán nuevas alternativas a los viejos problemas. Se dará un cambio fundamental en los sistemas de negocios y de información para utilizar estas nuevas tecnologías.

¡Una cosa *no* cambiará! Seguiremos necesitando analistas de sistemas que entiendan la forma de investigar y analizar a profundidad los problemas del negocio y definir los requerimientos lógicos como un requisito previo al diseño de sistemas. Pero todos tendremos que hacer eso con una velocidad y precisión cada vez mayores para cumplir con los programas de desarrollo de sistemas requeridos en la rápida economía del mañana.

Las habilidades interpersonales y de comunicaciones son esenciales para esta tarea. Las habilidades personales como la capacidad de vender y de persuadir se vuelven importantes. (Muchas escuelas ofrecen cursos de expresión oral y de comunicaciones en estos temas.) Los analistas de sistemas deben ser capaces de escribir un informe de negocios formal y de hacer una presentación sin entrar en temas o alternativas técnicas.

Esto concluye la fase de análisis de decisión. Y también nuestra cobertura de análisis de sistemas.

**FIGURA 4.22** Esquema de una propuesta de sistema típica

- I. Introducción
  - A. Propósito del informe
  - B. Antecedentes del proyecto que lleva a este informe
  - C. Alcance del proyecto
  - D. Estructura del informe
- II. Herramientas y técnicas utilizadas
  - A. Solución generada
  - B. Análisis de factibilidad (costo-beneficio)
- III. Requerimientos de sistemas de información
- IV. Soluciones alternativas y análisis de factibilidad
- V. Recomendaciones
- VI. Apéndices

# Mapa de aprendizaje

En este capítulo se proporcionó un panorama detallado de las fases de análisis de sistemas de un proyecto. Ahora está listo para aprender algunas de las habilidades de sistemas presentadas en este capítulo. Para la mayoría de los estudiantes, este sería el momento ideal para estudiar las técnicas de identificación de hechos que fueron identificados como críticos para casi cada fase y tarea descritos en este capítulo. En el capítulo 5 se enseñan estas habilidades. Se recomienda que usted lea el capítulo 6, “Modelado de requerimientos del sistema con los casos de uso”, antes de proceder a cualquiera de los capítulos de elaboración de modelos ya que los casos de uso son utilizados comúnmente para facilitar la actividad del modelado.

La secuencia de los capítulos de elaboración de modelos es flexible; sin embargo, personalmente preferimos y recomendamos que el capítulo 7, “Modelado y análisis de datos”, se estudie primero. Todos los sistemas de información incluyen bases de datos y la elaboración de modelos de datos es una habilidad esencial para el desarrollo de bases de datos. También, es más fácil sincronizar los modelos de datos iniciales con modelos de procesos posteriores que hacerlo al contrario. Su instructor podría preferir que usted primero estudie el capítulo 8, “Modelado de procesos”.

Si usted va directo al capítulo de elaboración de modelos de sistemas desde este capítulo, haga un compromiso de regresar al capítulo 5 para estudiar las técnicas de identificación de hechos. Sin importar qué tan bien domine la elaboración de modelos de sistemas, esa habilidad de modelado es completamente dependiente de su capacidad de identificar y reunir los hechos de negocios que subyacen en los modelos.

Para aquellos de ustedes que ya han terminado un curso de análisis de sistemas, este capítulo probablemente fue programado sólo como una revisión o contexto para el diseño de sistemas. Sugerimos que sólo revise los capítulos de modelado de sistemas y que proceda directamente al capítulo 10, “Diseño de sistemas”, el cual continuará donde termina este capítulo.

## Resumen

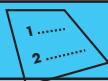


1. Formalmente, el análisis de sistemas es la disección de un sistema en sus componentes. Como una fase de solución de problemas, precede al diseño de sistemas. Con relación al desarrollo de sistemas de información, el análisis de sistemas es la investigación preliminar de un proyecto propuesto, el estudio y el análisis de problemas del sistema existente, el análisis de requerimientos de los requerimientos de sistemas para el nuevo sistema y el análisis de decisión para que las soluciones alternativas satisfagan los requerimientos.
2. Los resultados del análisis de sistemas se almacenan en un repositorio para su uso en las fases y proyectos posteriores.
3. Existen diversas estrategias populares o en surgimiento para el análisis de sistemas. Estas técnicas pueden ser utilizadas en combinación entre ellas.
  - a) Las técnicas de análisis basadas en modelos enfatizan el dibujo de modelos de sistemas pictóricos que representan una realidad actual o una visión objetivo del sistema.
  - i) El análisis estructurado es una técnica que se enfoca en la elaboración de modelos de procesos.

- ii) La ingeniería de información es una técnica que se enfoca en la elaboración de modelos de datos.
- iii) El análisis orientado a objetos es una técnica que se enfoca en elaboración de modelos de objetos que encapsulan las preocupaciones de datos y procesos que actúan en esos datos.
- b) Los enfoques de análisis acelerados enfatizan la construcción de modelos de sistemas de un sistema en un esfuerzo por acelerar el análisis de sistemas.
  - i) La elaboración de prototipos de identificación es una técnica que se enfoca en la construcción de subsistemas de pequeña escala, subsistemas funcionales para identificar los requerimientos.
  - ii) El análisis de arquitectura rápido intenta generar en forma automática los modelos de sistemas a partir de prototipos o de sistemas existentes. La generación automática de modelos requiere tecnología de ingeniería inversa.
  - c) Los enfoques basados en modelos y los de análisis de sistema acelerado dependen de las téc-

- nicas de identificación de requerimientos para identificar o extraer problemas y requerimientos de los propietarios y usuarios de sistemas.
- i) La identificación de hechos es el proceso formal de utilizar investigación, entrevistas, cuestionarios, muestreo y otras técnicas para reunir información.
  - ii) Las técnicas de planeación conjunta de requerimientos (JRP) utilizan talleres facilitados para reunir a todas las partes interesadas y acelerar el proceso de identificación de hechos.
- d) El rediseño de proceso de negocios es una técnica que se enfoca en simplificar y dirigir los procesos de negocios fundamentales antes de aplicar la tecnología de información a esos procesos.
4. Cada fase de análisis de sistemas (investigación preliminar, análisis de problemas, análisis de requerimientos y análisis de decisión) puede ser entendida en el contexto de los componentes del sistema de información: CONOCIMIENTO, PROCESOS Y COMUNICACIONES.
5. El propósito de la fase de investigación preliminar es determinar el valor del proyecto y crear un plan para completar esos proyectos considerados merecedores de un estudio y un análisis detallado. Para cumplir con la fase de investigación preliminar, el analista de sistemas trabajará con los propietarios y usuarios de sistemas para: a) Listar problemas, oportunidades y directrices; b) negociar el alcance preliminar; c) evaluar el valor del proyecto; d) planear el proyecto y e) presentar el proyecto a la comunidad de negocios. El producto de la fase de investigación preliminar es un diagrama del proyecto que debe ser aprobado por los propietarios de sistemas y/o un comité de toma de decisiones, comúnmente llamado comité de dirección.
6. El propósito de la fase de análisis del problema es responder las preguntas, ¿vale la pena resolver los problemas? y ¿vale la pena construir un nuevo sistema? Para responder estas preguntas, la fase de análisis del problema analiza profundamente los supuestos problemas y las oportunidades identificadas primero en la fase de investigación preliminar. Para completar la fase del análisis del problema, el analista continuará trabajando con el propietario del sistema, usuarios de sistemas y demás administración y personal de sistemas de información. El analista de sistemas y los participantes adecuados a) estudiarán el dominio del problema; b) analizarán profundamente los problemas y oportunidades; c) opcionalmente, analizarán los procesos de negocios; d) establecerán objetivos y restricciones de mejora de sistemas; e) actualizarán el plan de proyecto y f) presentarán los resultados y recomendaciones. El producto obtenido de la fase de análisis de problemas es el conjunto de objetivos de mejora de sistemas.
7. El propósito de la fase de análisis de requerimientos es identificar lo que deberá hacer el nuevo sistema sin la consideración de la tecnología; en otras palabras, definir los requerimientos de negocios para un nuevo sistema. Como en las fases de investigación preliminar y en la de análisis de problemas, el analista trabaja activamente con los usuarios y los propietarios de sistemas así como con otros profesionales de sistemas de información. Para completar la fase de análisis de requerimientos, el analista y los participantes adecuados a) definirán los requerimientos, b) analizarán los requerimientos funcionales por medio de la elaboración de modelos de sistemas o prototipos de identificación, c) rastrearán y completarán la definición de requerimientos, d) otorgarán prioridades a los requerimientos, y e) actualizarán el plan de proyecto y el alcance. El producto de la fase de análisis de requerimientos es la definición de requerimientos de negocios. Como los requerimientos son un objetivo móvil sin terminación, el análisis de requerimientos también incluye la tarea continua de administrar cambios a los requerimientos.
8. El propósito de la fase de diseño lógico es documentar los requerimientos de negocios por medio de modelos de sistemas para el sistema propuesto. Estos modelos de sistemas pueden, según la metodología, ser cualquier combinación de los modelos de proceso, modelos de datos y modelos de objeto. Los modelos describen diversos aspectos de nuestros componentes. De manera alternativa, los prototipos podrían ser construidos para "identificar requerimientos". Algunos prototipos de identificación pueden recibir una ingeniería inversa para convertirse en modelos de sistemas. El analista de sistemas y los participantes apropiados a) estructurarán o elaborarán un prototipo de requerimientos funcionales, b) validarán los requerimientos funcionales y c) definirán casos de aceptación de prueba. Estas tareas no son necesariamente secuenciales; pueden ocurrir en paralelo. El producto de la fase de diseño lógico es la definición de requerimientos de negocios.
9. El propósito de la fase de análisis de decisión es encaminar el proyecto partiendo de las necesidades del negocio hacia las soluciones, al identificar, analizar y recomendar una solución técnica para el sistema. Para completar la fase de análisis de decisión, el analista y los participantes adecuados a) definirán las soluciones alternativas; b) analizarán las soluciones alternativas para factibilidad (técnica, operativa, económica y de cronograma); c) compararán las soluciones alternativas factibles para elegir una o más soluciones recomendadas; d) actualizarán el plan de proyecto con base en la solución recomendada; y e) presentarán y defenderán la solución objetivo. El producto de la fase de análisis de decisión es la propuesta del sistema.

## Preguntas de repaso



1. ¿Cuáles son los factores del negocio que impulsan el análisis de sistemas? Con base en estos factores, ¿qué deben abordar los análisis de sistemas?
2. ¿Qué es el análisis basado en modelos? ¿Por qué se utiliza? Proporcione varios ejemplos.
3. ¿Cuál es el mayor enfoque del análisis estructurado?
4. ¿Cuál es el mayor enfoque de la ingeniería de información?
5. ¿Por qué el análisis orientado a objetos se ha vuelto popular? ¿Qué problemas resuelve?
6. ¿Cuáles son las cinco fases del análisis de sistemas?
7. ¿Cuál es la meta de la fase de definición de alcance?
8. ¿Cuáles son las cinco tareas que usted hace en la fase de definición de alcance?
9. ¿Cuál es el gatillo para comunicar el plan del proyecto y a quiénes se comunicará? ¿Por qué es importante comunicar el plan de proyecto?
10. ¿Por qué muchos analistas de sistemas novatos fallan en analizar eficazmente los problemas? ¿Qué pueden hacer ellos para volverse más eficientes?
11. ¿Cuál es una herramienta popular utilizada para identificar y expresar los requerimientos funcionales de un sistema?
12. ¿Cuál es una técnica comúnmente utilizada para asignar prioridades a los requerimientos de sistemas?
13. ¿Cuándo podría utilizarse la elaboración de prototipos en lugar de la de modelos de sistemas para determinar los requerimientos funcionales?
14. ¿Por qué se necesita la fase de análisis de decisión?
15. ¿Cuáles son algunas formas para identificar las soluciones alternativas?

## Problemas y ejercicios



1. Hay muchos enfoques distintos para el análisis de sistemas. A pesar de estos enfoques distintos, ¿cuál es la definición universalmente aceptada del análisis de sistemas? ¿Cuál es la opinión general acerca de cuándo comienza el análisis de sistemas y cuándo termina? Como administrador de proyecto, ¿qué es importante saber acerca de la definición del análisis de sistemas y qué es importante asegurar en su organización en cuanto a la definición?
2. Como un analista de sistemas, usted estará expuesto a utilizar muchos enfoques distintos para el

análisis de sistemas a lo largo de su carrera. Es importante que entienda la base conceptual de cada tipo de método, sus diferencias, fortalezas y debilidades esenciales. Considere las diferencias en el análisis estructurado, la ingeniería de información y la elaboración de modelos de datos, así como el análisis orientado a objetos, todos los cuales representan un análisis basado en modelos y conteste la matriz que se presenta a continuación.

	CENTRICIDAD (datos, proceso, etcétera)	TIPO DE MODELOS UTILIZADOS	DIFERENCIAS ESENCIALES
ANÁLISIS ESTRUCTURADO			
INGENIERÍA DE INFORMACIÓN Y ELABORACIÓN DE MODELOS DE DATOS			
ANÁLISIS ORIENTADO A OBJETOS			

3. Los métodos de análisis de sistemas acelerados están basados en la premisa de que los prototipos pueden ayudar a revelar los requerimientos de negocios más importantes, mucho más rápido que los otros métodos. Describa los dos métodos más comúnmente utilizados para el análisis acelerado. ¿Qué hacen y cómo lo hacen? ¿Cuál es una de las críticas de la elaboración de prototipos? ¿Los métodos de análisis de sistemas acelerados reemplazan completamente a los métodos más formales, tales como un análisis estructurado?
4. Durante la fase de definición de alcance, ¿cuál es una pregunta que usted nunca debe perder de vista? ¿Y cómo la responde usted? ¿Cuáles cinco tareas deben ocurrir durante la fase de definición de alcance?
5. Usted es un analista de sistemas novato y está ansioso de probar sus capacidades en su primer proyecto. Usted está en una junta de análisis de problemas con los propietarios y los usuarios de sistemas y dice, "necesitamos hacer esto para resolver el problema". ¿En qué trampa común está en peligro de caer? ¿Qué técnica podría utilizar para evitar esta trampa?
6. Su equipo de proyecto ha completado la fase de definición de alcance y está ahora en el punto en la fase de análisis de problemas para establecer los objetivos de mejora de sistemas. Como analista de sistemas en el equipo de proyecto, usted es el facilitador de una sesión de lluvia de ideas para definir los objetivos de mejora de sistemas. Como varios de los propietarios y usuarios del proyecto nunca han hecho esto antes, describa las características de los objetivos de mejora de sistemas adecuados y proporcione algunos ejemplos. Los miembros del equipo de proyecto sugieren los siguientes objetivos:
  - a) Reduzca el tiempo requerido para procesar el pedido.
  - b) El nuevo sistema debe utilizar Oracle para almacenar datos.
  - c) Las pantallas de entrada de datos deben ser rediseñadas para que sean más amigables al usuario.
  - d) La tasa de satisfacción del cliente con el proceso de pedido en línea debe aumentarse en un 10 por ciento.¿Son estos ejemplos de objetivos de mejora de sistemas adecuados? ¿Por qué sí o por qué no? Si no, ¿cómo se pueden replantear? Frecuentemente, los objetivos tienen restricciones que están vinculadas a ellos; ¿cuál cree usted que sería la restricción que concuerda con estos dos objetivos?
7. Usted ha pasado ya por la fase de análisis de problemas del proyecto, y ahora comienza la fase de análisis de requerimientos. Durante la primera reunión acerca de los requerimientos del negocio, uno de los otros analistas en el equipo del proy-  
ecto pregunta a los usuarios de sistemas, "¿cómo debe satisfacer las necesidades del negocio el nuevo sistema?" ¿Qué error común comete el analista de sistemas? ¿Cuáles son con frecuencia las consecuencias de cometer este error?
8. ¿Cuál es la diferencia entre requerimientos funcionales y no funcionales y cuál es el propósito de clasificarlos en estas categorías? ¿Cuáles son dos formatos que un analista puede utilizar para documentar los requerimientos de sistema funcionales?
9. ¿Es importante priorizar los requerimientos de sistemas? y si es así, ¿cuándo debe hacerse esto? ¿Cuál es una técnica que puede ser utilizada y cuál es la diferencia entre requerimientos obligatorios y deseables? ¿Cuál es una forma de probar si un requerimiento es realmente obligatorio?
10. Una vez que se identifican los requerimientos del sistema y que se asignan las prioridades, ¿no deberían limitarse los requerimientos para prevenir un aumento del alcance o de las características? ¿El actualizar el plan del proyecto o permitir a los interesados continuar solicitando cambios sólo retrasará el diseño y la construcción del sistema e incluso hasta la terminación del proyecto mismo?
11. ¿Por qué los casos de aceptación de pruebas deben definirse durante la fase de diseño lógico? Después de todo, el diseño técnico no ha sido hecho aún, mucho menos la construcción del sistema. ¿No deberían esperar las actividades de prueba al menos hasta que la construcción esté en proceso?
12. ¿En qué difiere la fase de diseño lógico de la fase de análisis de requerimientos?
13. Digamos que usted forma parte del equipo de proyecto de un sistema que ha tenido muchos problemas durante la fase de análisis de requerimientos y que lleva varias semanas de retraso. El administrador del proyecto quiere tratar de recuperar el tiempo perdido al pasar por alto algunas tareas o abreviarlas en la fase de diseño lógico. El administrador del proyecto argumenta que, después de todo, ahora tienen una idea clara de los requerimientos, los diseñadores y los constructores son realmente experimentados y no necesitan el diseño lógico con el fin de hacer el diseño técnico. ¿Este es un método legítimo de ponerse al corriente con el cronograma de trabajo? ¿Cuáles son las posibles consecuencias?
14. Al identificar y definir las soluciones alternativas posibles, ¿cuáles son los papeles típicos de los diversos interesados que participan en el proyecto?
15. Usted es un analista de sistemas y se le ha pedido revisar la factibilidad del análisis y la evaluación de diversas soluciones de sistemas. ¿Qué conjuntos de criterios utilizaría normalmente? ¿Debe usted comparar las soluciones alternativas entre ellas en este punto? ¿Por qué sí o por qué no? ¿Cuál es el producto típico que resulta de esta tarea?

## Proyectos e investigación



1. Elija un sistema de información con el que esté familiarizado y que sienta debe ser mejorado, con base en sus experiencias como empleado, cliente, usuario de otro sistema o propietario de sistemas. Cambie de roles y de perspectivas conforme sea necesario realizar o responder lo siguiente:
  - a) Describa la naturaleza del sistema de información que ha elegido.
  - b) Describa a la organización que es dueña y mantiene el sistema de información.
  - c) Identifique los problemas y oportunidades de la línea de base, por la tarea 1.1.
  - d) Desarrolle una definición de problema preliminar, por medio del formato mostrado en la figura 4.8.
2. Asuma que usted es ahora un analista de sistemas en el proyecto descrito en la pregunta anterior. La administración ejecutiva estuvo realmente impresionada por su trabajo en la definición del problema. Como resultado, han dado el “visto bueno” al proyecto, se ha desarrollado el cronograma de trabajo y el de presupuesto, así mismo, el plan del proyecto ha sido aprobado por el comité de dirección ejecutiva. Como el analista de sistemas, ahora se le han asignado las siguientes tareas:
  - a) Desarrollar y documentar su comprensión sobre el dominio y vocabulario de negocios de este problema, utilizando el marco de referencia de componentes de sistemas de información de este libro, tal como se describe en la tarea 2.1.
  - b) Analizar los problemas y oportunidades por medio de un análisis de causa y efecto (tarea 2.2).
  - c) Analizar los procesos de negocios y desarrollar los modelos de proceso (tarea 2.3).
  - d) Establecer los objetivos de mejora de sistemas (tarea 2.4).
  - e) Preparar una matriz de problemas, oportunidades, objetivos y restricciones, utilizando la figura 4.12 como un ejemplo.
3. Comunicar los resultados y recomendaciones es la tarea final en la fase de análisis de problemas. Como analista de sistemas en el proyecto, usted tiene la tarea de preparar el informe de objetivos de mejora de sistemas y recomendaciones. Para este ejercicio, prepare sólo la porción de resumen ejecutivo del informe, al usar el formato mostrado en la figura 4.13. El comité de dirección ejecutivo utilizará este resumen para tomar sus decisiones en relación con las recomendaciones.
4. Hasta el momento, su trabajo eficiente en el proyecto ha continuado impresionando a la administración ejecutiva. Usted ha recibido un aumento de sueldo y se le ha pedido conducir la fase de análisis de requerimientos. Específicamente:
  - a) Identificar los requerimientos de sistemas y preparar un esquema de requerimientos funcionales y no funcionales, de acuerdo a la tarea 3.1. Como su organización utiliza la técnica de análisis estructurado y no la de modelo de casos, liste cada objetivo de mejora de sistemas, las entradas, los procesos, las salidas y los datos almacenados requeridos para cumplir con cada objetivo.
  - b) Asuma que los requerimientos que usted identificó en el paso anterior han sido validados. Asigne prioridades a los requerimientos de acuerdo con su importancia relativa y utilice el método descrito en la tarea 3.2.
5. Su trabajo ha ayudado a mantener el proyecto por delante del programa, así que la administración ejecutiva le da un par de semanas de vacaciones pagadas. Cuando usted regresa, el proyecto va hacia la fase de análisis de decisión. Su próxima tarea es identificar las soluciones alternativas.
  - a) Describa el proceso para identificar las soluciones alternativas. ¿Qué debe tener cuidado de *no* hacer en este punto?
  - b) Desarrolle una matriz de soluciones alternativas del sistema y utilice el formato de la figura 4.20 como ejemplo e incluya tres posibles soluciones.
6. Después de identificar soluciones alternativas, el siguiente paso es analizarlas.
  - a) Describa el proceso para analizar soluciones alternativas. ¿Qué es lo que *no* debe hacer el equipo del proyecto para completar esta tarea?
  - b) Desarrolle una matriz de análisis de factibilidad con base a las soluciones alternativas identificadas en la pregunta anterior y utilice el formato que se muestra en la figura 4.21 como un ejemplo. Determine los factores de peso para considerar cada solución.



1. Usted es el director de información de un comercio importante. Recientemente, usted leyó "Spying on the Sales Floor" en el *Wall Street Journal* el 21 de diciembre de 2004. Usted observa que sus competidores utilizan la revisión y exploración de videos para analizar el comportamiento de los consumidores. ¿Debe adoptar su compañía esta herramienta (revisión de videos)? ¿Cuáles son las implicaciones estratégicas para su compañía que tiene el movimiento de sus competidores? ¿Qué oportunidades han sido creadas? ¿Amenazas?
2. Lea "Human Reengineering" de Cooper y Markus, en la *Sloan Management Review*, verano de 1995. En este artículo, Okuno trabaja para instituir una actitud positiva hacia el cambio. ¿Cómo hace él esto? Analice la importancia de la aceptación del cambio por parte de los empleados para alcanzar el éxito de la implementación de tecnología.
3. Refiérase al caso breve número 1. Usted, como director de información, cree que las ganancias por implementar la exploración de los videos en sus tiendas minoristas sobrepasarán cualquier percepción negativa de sus clientes. Su compañía es Baby's R Us, una compañía filial de Toys R Us. Realice un estudio de factibilidad para esta inversión. Asegure incluir un listado de costos y beneficios intangibles, así como una razón para elegir una tasa de descuento. ¿Cuál es el retorno de inversión al utilizar la exploración de videos? Trate de elaborar su análisis usando como máximo 15 páginas.
4. Desarrolle un plan de proyecto y un estudio de factibilidad de programa para la inversión de la estrategia de revisión o exploración de videos en Baby's R Us. Asegure incluir un diagrama de Gantt y PERT/CPM, así como un análisis claro de todas las tareas que deberán completarse.



## Ejercicios de equipo e individuales

1. ¿Con qué frecuencia cree usted que los temas legales juegan un papel importante en el éxito del proyecto? Piense en un ejemplo de un sistema de información potencialmente bueno o un programa que haya sido restringido o que no haya sido factible debido a los requerimientos legales.
2. Como equipo, realice una lluvia de ideas sobre formas de mejorar la aceptación del cambio por parte

de los empleados que utilizan nuevos sistemas de información o procesos de negocios.

3. Piense un ejemplo de cuando la mejora de procesos del negocio es más apropiada que la reingeniería de proceso de negocios. Comparta su opinión con la clase.



## Lecturas recomendadas

*Application Development Trends* (periódico mensual). Natick, MA: Software Productivity Group, una compañía de ULLO International. Éste es nuestro periódico favorito de desarrollo de sistemas. Sigue al análisis de sistemas y a las estrategias de diseño, metodologías, CASE y otras tendencias relevantes. Visite su sitio Web en [www.adtmag.com](http://www.adtmag.com).

Grause, Donald C. y Gerald M. Weinberg. *Are Your Lights On? How to Figure Out What the Problem REALLY Is*, Nueva York: Dorset House Publishing, 1990. Aquí hay un título que realmente debe hacerlo pensar y el libro completo aborda uno de los aspectos menos publicados del análisis de sistemas: solución de problemas.

Hammer, Mike, "Reengineering Work: Don't Automate, Obliterate", *Harvard Business Review*, julio-agosto de 1990, pp. 104-111. El Dr. Hammer es un conocido experto en el rediseño de procesos de negocios. Este escrito inicial analiza algunos casos clásicos donde la técnica agregó valor en forma dramática al negocio.

Whetherbe, James, *Systems Analysis and Design: Best Practices*, 4a. ed., St. Paul, MN: West Publishing, 1994. Estamos en deuda con el Dr. Wetherbe por el marco de referencia PIECES.

Wood, Jane y Denise Silver, *Joint Application Design: How to Design Quality Systems in 40% Less Time*. Nueva York, John Wiley & Sons, 1989. Este libro proporciona una presentación profunda excelente del desarrollo de aplicación conjunta (JAD).

Youardon, Edward, *Modern Structured Analysis*, Englewood Cliffs, NJ, Yourdon Press, 1989. Esta es una actualización al texto clásico de DeMarco acerca del mismo tema el cuál define el estado actual de la práctica para el método de análisis estructurado.

Zachman, John, A., "A Framework for Information System Architecture", *IBM Systems Journal*, 26, núm. 3, 1987. Este artículo presenta un marco de referencia conceptual popular de inspección de sistemas de información, así como el desarrollo de una arquitectura de información.



# Tema 5

Técnicas de exploración y  
análisis de requerimientos



# 5 Técnicas de exploración de hechos para identificación de los requerimientos

Las técnicas efectivas para explorar hechos son cruciales en el desarrollo de los proyectos de sistemas. En este capítulo aprenderá sobre las técnicas para identificar y analizar requerimientos de sistemas de información. Usted aprenderá cómo usar varias técnicas de identificación para reunir información acerca de los problemas, oportunidades y directivas del sistema. Usted sabrá que ha comprendido las técnicas de exploración de hechos e identificación de requerimientos cuando pueda:

- Definir los requerimientos del sistema y diferenciar entre los requerimientos funcionales y los no funcionales.
- Comprender la actividad de análisis del problema y ser capaz de crear un diagrama Ishikawa (de espina de pescado) para ayudar a resolver el problema.
- Comprender el concepto de administración de requerimientos.
- Reconocer siete técnicas para explorar hechos y caracterizar las ventajas y desventajas de cada uno.
- Comprender seis guías para escuchar efectivamente.
- Comprender qué preocupa por eso.
- Describir a los participantes típicos en una sesión o planeación conjunta de requerimientos (JRP) y describir sus roles.
- Completar el proceso de planeación con una sesión JRP, que incluya seleccionar y equipar el local, seleccionar a los participantes, y preparar una agenda para guiar la sesión JRP.
- Describir varios beneficios de usar JRP como una técnica de identificación de hechos.
- Describir una estrategia que explore hechos para sacar el mejor provecho de su tiempo con los usuarios finales.

## Introducción

Bob Martínez ha dedicado la mayor parte de la semana a leer. Comenzó con memorandos relacionados con el Sistema de Servicios a Miembros para comprender mejor el problema. Entonces revisó el manual de procedimientos de SoundStage para cualquier plan relacionado a los servicios a clientes y promociones. Estudió casi 100 formatos de solicitudes de miembros seleccionadas al azar, notando los tipos de información registrados en cada espacio y cuáles espacios fueron ocupados siempre, cuáles algunas veces y los que en ninguna ocasión fueron empleados. Él leyó la documentación del actual sistema de servicios a miembros de SoundStage. Revisó la información y los diagramas del anterior proyecto de desarrollo de sistemas de servicio tomando en cuenta las cosas que probablemente necesitarían ser cambiadas en el nuevo sistema. Fue un trabajo agotador. Pero al final realmente sintió que estaba empezando a comprender el sistema. Elaboró un reporte para Sandra, su jefa, acerca de los temas principales y las preguntas que necesitarían ser contestadas en la próxima reunión de planeación conjunta de requerimientos.

## Una introducción a la identificación de requerimientos

En el capítulo 3 analizamos varias fases del desarrollo de sistemas. Cada una es importante y necesaria para efectivamente diseñar, construir y, en última instancia, implementar un sistema que esté a la altura de las necesidades de los involucrados.\* Pero para desarrollar dicho sistema, debemos primero ser capaces de identificar correctamente, analizar y entender cuáles son los requerimientos de los usuarios o lo que los usuarios quieren que haga el sistema. El proceso y las técnicas que un analista de sistemas usa para identificar, analizar y entender requerimientos de sistema, son referidos como la **identificación de requerimientos**. Tal como ha sido sugerido en la página inicial del capítulo, la identificación de los requerimientos involucra principalmente a los analistas de sistemas que trabajan con los usuarios de sistemas y con los propietarios durante las primeras fases de desarrollo del sistema, con el fin de obtener una comprensión detallada de los requerimientos del negocio de un sistema de información.

¿Qué son los requerimientos del sistema? Los **requerimientos del sistema** especifican lo que el sistema de información deberá hacer o cuál propiedad o calidad debe de tener éste. Los requerimientos del sistema que especifican lo que el sistema de información debe hacer son frecuentemente llamados **requerimientos funcionales**. Aquellos que especifican una propiedad o calidad que el sistema debe tener con frecuencia son llamados **requerimientos no funcionales**.

El marco de referencia PIECES (tabla 5.1), presentado en el capítulo 3, proporciona una herramienta excelente para la clasificación de los requerimientos del sistema. La ventaja de clasificar los varios tipos de requerimientos es la habilidad de agrupar requerimientos para reportar, rastrear y validar los propósitos. Hacer eso ayuda a identificar los requerimientos que tal vez han sido pasados por alto.

Esencialmente, el propósito de la identificación de los requerimientos y su administración es identificar correctamente los requerimientos de CONOCIMIENTO, PROCESO y COMUNICACIÓN para los usuarios de un sistema nuevo. La falla de identificar correctamente los requerimientos del sistema puede dar como resultado en una o más de las siguientes:

- El sistema puede costar más de lo proyectado.
- El sistema puede ser entregado después de lo prometido.
- El sistema puede no estar a la altura de las expectativas de los usuarios, y esa insatisfacción puede originar que no lo usen.
- Una vez en producción, los costos de mantenimiento y mejora del sistema pueden ser excesivamente altos.
- El sistema puede ser poco confiable y tener la tendencia a fallar y tener mucho tiempo muerto.
- La reputación del equipo de TI en el grupo se mancha debido a cualquier falla, sin importar quién ha cometido el error, dicha falla será percibida como un error del equipo.

\* Nota del R.T.: En inglés se les llama *stakeholders* a las personas que serán afectadas de uno u otro modo por el sistema.

**TABLA 5.1 PIECES Clasificación de los requerimientos del sistema**

Tipo de requerimiento no funcional	Explicación
Desempeño	<p>Los requerimientos de desempeño representan el desempeño que el sistema debe tener para satisfacer las necesidades de los usuarios.</p> <ul style="list-style-type: none"> <li>• ¿Cuál es el ritmo aceptable de producción?</li> <li>• ¿Cuál es el tiempo aceptable de respuesta?</li> </ul>
Información	<p>Los requerimientos de información representan la información que es útil al usuario en cuanto a contenido, tiempo, exactitud y formato.</p> <ul style="list-style-type: none"> <li>• ¿Cuáles son las entradas y salidas necesarias? ¿Cuándo deben suceder?</li> <li>• ¿Cuáles son los datos requeridos que deben almacenarse?</li> <li>• ¿Qué tan actualizada debe estar la información?</li> <li>• ¿Cuáles son las interfaces con los sistemas externos?</li> </ul>
Economía	<p>Los requerimientos de ahorro representan la necesidad de que el sistema reduzca costos o incremente ganancias.</p> <ul style="list-style-type: none"> <li>• ¿Cuáles son las áreas del sistema donde los costos deben reducirse?</li> <li>• ¿Cuánto deberían reducirse los costos o incrementarse las ganancias?</li> <li>• ¿Cuáles son los límites del presupuesto?</li> <li>• ¿Cuál es el cronograma del desarrollo?</li> </ul>
Control (y seguridad)	<p>Los requerimientos de control representan el ambiente en el cual el sistema debe operar, así como el tipo y grado de seguridad que debe alcanzarse.</p> <ul style="list-style-type: none"> <li>• ¿Debe controlarse el acceso al sistema o a la información?</li> <li>• ¿Cuáles son los requerimientos de privacidad?</li> <li>• La importancia de la información, ¿necesita de un manejo especial (respaldo, almacenaje fuera del lugar, etcétera)?</li> </ul>
Eficiencia	<p>Los requerimientos de eficiencia representan la capacidad del sistema para producir salidas con mínimo desperdicio.</p> <ul style="list-style-type: none"> <li>• ¿Hay pasos duplicados en el proceso que deban eliminarse?</li> <li>• ¿Hay formas de reducir el desperdicio por la manera en que el sistema usa sus recursos?</li> </ul>
Servicio	<p>Los requerimientos de servicio representan necesidades que funcionen para que el sistema sea confiable, flexible y expandible.</p> <ul style="list-style-type: none"> <li>• ¿Quiénes usarán el sistema, y dónde están localizados?</li> <li>• ¿Habrá distintos tipos de usuarios?</li> <li>• ¿Cuáles son los factores humanos adecuados?</li> <li>• ¿Qué dispositivos y materiales de entrenamiento deben incluirse en el sistema?</li> <li>• ¿Qué dispositivos y materiales de entrenamiento deben ser desarrollados y mantenidos separadamente del sistema, en programas y bases de datos de entrenamiento independientes basado en computadora (CBT)?</li> <li>• ¿Cuáles son los requerimientos de confiabilidad/disponibilidad?</li> <li>• ¿Cómo deberá ser empacado y distribuido el sistema?</li> <li>• ¿Qué documentación se necesita?</li> </ul>

**TABLA 5.2 Costos relativos para corregir un error**

<b>Etapa en la cual se detecta el error</b>	<b>Relación de costos</b>
Requerimientos	1
Diseño	3-6
Codificación	10
Desarrollo de pruebas	15-40
Pruebas de aceptación	30-70
Operación	40-1 000

El impacto en los costos puede ser impresionante. Consideré por ejemplo la tabla 5.2, de Barry W. Boehm, un experto en economía de la tecnología de información.<sup>1</sup> Él estudió varios proyectos de desarrollo de software para determinar los costos de los errores en los requerimientos que fueron descubiertos más tarde en el proceso de desarrollo.

Basado en los hallazgos de Boehm, un requerimiento equivocado que no fue detectado y arreglado hasta la fase de la operación, costará 1 000 veces más de lo que costaría si fuera detectado y fijado en la fase de requerimientos. Por tanto, al definir los requerimientos del sistema, es muy importante que éstos obedezcan los siguientes lineamientos:

- *Consistentes*: Los requerimientos son no contradictorios o ambiguos.
- *Completos*: Los requerimientos describen todas las posibles entradas del sistema y las respuestas.
- *Factibles*: Los requerimientos pueden satisfacerse con los recursos disponibles y sus restricciones (el análisis de factibilidad se cubre en el capítulo 9).
- *Requeridos*: Los requerimientos se necesitan en realidad y cumplen con el objetivo del sistema.
- *Exactos*: Los requerimientos son expresados de manera correcta.
- *Rastreables*: Los requerimientos apuntan directamente hacia las funciones y características del sistema.
- *Verificables*: Los requerimientos se definen de forma en que pueden ser comprobados durante la prueba.

Este proceso puede ser difícil, frustrante y tomar mucho tiempo; por ello a menudo las empresas y los individuos toman atajos para ahorrar tiempo y dinero. Sin embargo, este criterio tan estrecho a menudo conduce a los problemas mencionados anteriormente. Ahora que comprendemos nuestro objetivo, analicemos el proceso.

## Proceso de identificación de requerimientos

El proceso de identificación de requerimientos consiste de las siguientes actividades:

- Identificación del problema y análisis.
- Identificación de los requerimientos.
- Documentación y análisis de los requerimientos.
- Administración de los requerimientos.

Ahora examinemos cada una de estas actividades en detalle:

### > Identificación del problema y análisis

Tal y como fue previamente asentado, los requerimientos resuelven problemas. Para que los analistas de sistemas sean exitosos, deberán tener habilidades para analizar problemas. Para

<sup>1</sup> Donald C. Gause y Gerald M. Weinberg, *Exploring Requirements: Quality before Design* (Nueva York: Dorset House Publishing, 1989), pp. 17-18.

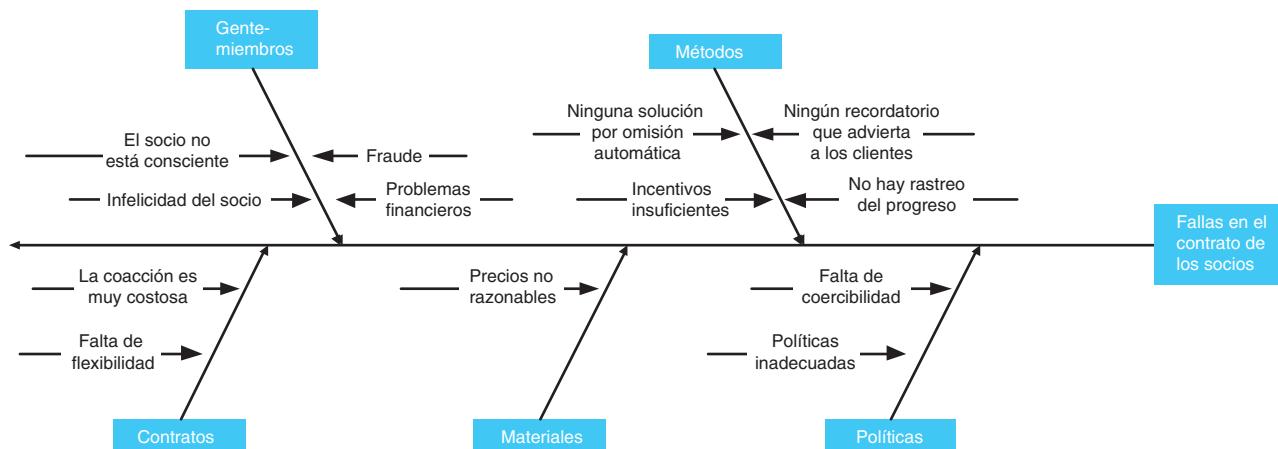
entender por completo el análisis de problemas, usemos el siguiente ejemplo: Una mujer lleva a su hijita al médico porque la niña está enferma. Lo primero que el médico intenta hacer es identificar el problema. La niñita tiene dolor de oídos, fiebre y catarro. ¿Son éstos los problemas? La madre ha estado dando a la niña medicinas contra el dolor, pero la niña no se ha mejorado. Lo que sucede es que la madre está dando tratamiento para los síntomas y no para el problema real. Afortunadamente, el doctor está capacitado para analizar más allá. Despues de examinar a la niña, el médico ha concluido que tiene una infección del oído, lo cual constituye la raíz de la causa de los síntomas. Ahora que el problema ha sido identificado y analizado, es tiempo de que el doctor ofrezca una solución. Normalmente, un antibiótico se prescribe para curar una infección de oído (otitis), pero para poder hacer eso, el doctor primero necesita determinar si existen contraindicaciones en la medicina que puede prescribir. ¿Qué edad tiene la niña y cuánto pesa? ¿La niña es alérgica a algún medicamento? ¿Puede tragar pastillas? Una vez que estos datos son conocidos, se puede recetar. Los analistas de sistemas aplican el mismo proceso de solución de problemas tal como lo usa un doctor, sin embargo en lugar de diagnosticar problemas médicos diagnostican problemas de sistemas.

Uno de los errores más comunes cometidos por analistas de sistemas inexpertos al tratar de analizar problemas es identificar un síntoma como un problema. Como resultado, pueden diseñar e implementar una solución que probablemente no resuelva el problema real o que pueda causar nuevos problemas. Una herramienta popular usada por equipos en desarrollo para identificar, analizar y resolver problemas, es un **diagrama Ishikawa**. El diagrama con forma de espina de pescado es el producto de Kaoru Ishikawa, pionero en la administración de procesos de calidad de los astilleros en Kawasaki, Japón. Por ello, se convirtió en uno de los padres fundadores de la administración moderna.

Al dibujar el diagrama de espina de pescado, se comienza con el nombre del problema que nos interesa. Éste se coloca del lado derecho del diagrama (o la *cabeza del pescado*). Las posibles causas del problema se dibujan como *espinas* fuera de la *columna vertebral principal*, cada una con una flecha que señale hacia la columna. Típicamente, estas “espinas” tienen cuatro categorías básicas: materiales, máquinas, mano de obra y métodos (las cuatro emes por materials, machines, manpower, y methods). Pueden adaptarse otros nombres al problema en cuestión. Otras categorías alternativas o adicionales incluyen posiciones, procedimientos, políticas y personas (las cuatro pes por places, procedures, policies, y people) o contexto, proveedores, sistemas y habilidades (las cuatro eses por surroundings, suppliers, systems, y skills).

La clave es tener de tres a seis categorías principales que abarquen todas las posibles áreas de causas. Comúnmente se llevan a cabo técnicas de lluvias de ideas (definidas posteriormente en este capítulo) para añadir causas a las espinas principales. Cuando se termina la lluvia de ideas, la espina de pescado ilustra una idea completa de todas las posibilidades acerca de lo que podría ser la causa de raíz para el problema encontrado. El equipo de desarrollo puede usar el diagrama para decidir y concordar en cuáles son las causas más probables del problema y cómo debería actuarse sobre ellas. La figura 5-1 es un ejemplo de un diagrama de espina de pescado que ilustra el problema de los miembros

**diagrama Ishikawa** Herramienta gráfica usada para identificar, explorar e ilustrar problemas, así como las causas y efectos de esos problemas. Es frecuente que se le llame diagrama de causas y efectos o diagrama de espina de pescado (porque se asemeja al esqueleto de un pez).



**FIGURA 5.1** Muestra de un diagrama de espina de pescado

de SoundStage que no cumplen con los contratos. En el diagrama, observe que el problema que debe resolverse está en el cuadro de la extrema derecha. Las cinco áreas que se han identificado como categorías de causas (Personas-miembros, Métodos, Contratos, Materiales, y Políticas) se listan en los cuadros arriba y abajo del *esqueleto de pescado* y se conectan con flechas (espinas) que apuntan a la columna vertebral del pescado. Las causas reales del problema para cada categoría se ilustran como flechas que apuntan hacia la flecha de la categoría (espina).

### > Identificación de los requerimientos

**exploración** Proceso formal del uso de la investigación, juntas, entrevistas, cuestionarios, muestreo, y otras técnicas para recabar información acerca de los problemas, los requerimientos, y las preferencias del sistema. También se llama *recopilación de información o recolección de datos*.

Dada una comprensión de los problemas, el analista de sistemas puede comenzar a definir los requerimientos. Para que los analistas de sistemas de la actualidad tengan éxito en definir los requerimientos de los sistemas, deben tener habilidad en los métodos efectivos para recopilar información: la exploración. La **exploración** es una técnica que se usa a través del ciclo completo de desarrollo, pero es muy crítica en la fase de análisis de requerimientos. Una vez que se ha terminado la exploración, se usarán herramientas tales como los casos de uso (prácticos), los modelos de datos, los modelos de procesos, y los modelos de objetos para documentar los hechos, y se obtendrán conclusiones a partir de estos hechos. Usted va a aprender acerca de estas herramientas y también cómo documentar los requerimientos derivados de la exploración en los capítulos subsiguientes de este libro.

Los hechos están en el dominio de la aplicación de negocios y de los usuarios finales. Por tanto, el analista debe recolectar estos hechos con objeto de aplicar con efectividad las técnicas y las herramientas de la documentación. Durante la fase del análisis de sistemas, el analista aprende acerca del vocabulario, los problemas, las oportunidades, las restricciones, los requerimientos, y las prioridades de un negocio y de un sistema.

¿Qué tipo de hechos deben recolectarse? Ciertamente sería benéfico si tuviéramos un marco de referencia que nos ayude a determinar qué hechos es necesario recolectar, independientemente del proyecto en el cual estemos trabajando. Afortunadamente, tenemos este marco. Tal como resulta, los hechos que describen a cualquier sistema de información también corresponden muy bien a los bloques de construcción que se realzan en la página principal del capítulo. Observe que las técnicas de exploración se usan en el desarrollo inicial de sistemas para identificar los alcances y la visión de la información, la funcionalidad, y la comunicación, así como para identificar el proceso de conocimiento del negocio, y los requerimientos de comunicaciones del sistema.

### > Requerimientos de documentación y de análisis

Cuando el analista de sistemas realiza actividades de exploración, es importante que reúna o documente la información recopilada (es decir, los *requerimientos del borrador*) de una manera organizada, inteligible y significativa. Estos documentos iniciales van a suministrar la dirección de las técnicas de modelación que el analista de sistemas va a usar para analizar los requerimientos y determinar los requerimientos correctos del proyecto. Una vez que éstos han sido identificados, el analista de sistemas formaliza los requerimientos presentándolos en un documento que será revisado y aprobado por los usuarios.

**Documentación de los requerimientos del borrador** Los analistas de sistemas usan diferentes herramientas para documentar sus identificaciones iniciales en forma de borrador. Ellos escriben los *casos de uso* para describir las funciones del sistema desde la perspectiva de los usuarios externos y de una manera y con una terminología que el usuario entienda. Se usan *tablas de decisión* para documentar las complejas políticas de negocios de una organización y las reglas para tomar decisiones, y se usan *tablas de requerimientos* para documentar cada requerimiento específico. Cada una de estas herramientas se examina con más detalle posteriormente en el capítulo.

**Análisis de los requerimientos** Muy frecuentemente, las actividades de exploración producen requerimientos que entran en conflicto entre sí. Esto se debe a que los requerimientos son solicitados por muchas fuentes diferentes y cada persona tiene sus propias opiniones y deseos acerca de la funcionalidad y las características del nuevo sistema. El objetivo de la actividad del análisis de requerimientos es identificar y resolver los problemas con los requerimientos y alcanzar un consenso sobre cualesquier modificaciones que satisfagan a los involucrados. El proceso tiene que ver con los requerimientos “iniciales” recopilados de los involucrados. Generalmente estos requerimientos

están incompletos y se documentan de una manera informal en instrumentos tales como los casos de uso, las tablas, y los reportes. El enfoque de esta etapa radica en alcanzar consensos sobre las necesidades de los involucrados; en otras palabras, el análisis debe responder a la pregunta: “¿Tenemos los requerimientos correctos del sistema para el proyecto?” Es inevitable que los requerimientos del borrador contengan muchos problemas, tales como:

- Requerimientos faltantes
- Requerimientos en conflicto
- Requerimientos infactibles
- Traslape de requerimientos
- Requerimientos ambiguos

Estos tipos de problemas de requerimientos son muy comunes en muchos de los documentos de requerimientos que se escriben actualmente. Si se dejan sin resolver, puede ser muy costoso repararlos posteriormente en el ciclo de desarrollo.

Anteriormente se mencionó que los involucrados deben concordar en los requerimientos resultantes del sistema: por tanto hay un proceso inevitable de negociación entre los involucrados durante el análisis. Si los involucrados múltiples proponen requerimientos que están en conflicto mutuo o si los requerimientos propuestos son demasiado ambiciosos, los involucrados deben negociar, frecuentemente bajo la guía del analista de sistemas, para concordar sobre cualesquiera modificaciones o simplificaciones de los requerimientos del sistema. También deben concordar sobre la condición crítica y la prioridad de los requerimientos. Esto es crucial para asegurar el éxito del esfuerzo de desarrollo.

Las actividades de exploración y del análisis de requerimientos están asociadas muy cercanamente entre sí y de hecho con frecuencia están entrelazadas. Si se encuentra que los requerimientos descubiertos durante el proceso de exploración son problemáticos, el analista puede seguir adelante y realizar actividades de análisis sobre los elementos seleccionados con objeto de resolver los problemas antes de seguir provocando necesidades y deseos adicionales del sistema.

Este capítulo se centra principalmente en el lado de los negocios de los requerimientos o, en otras palabras, los requerimientos lógicos, pero es importante observar que existen requerimientos técnicos adicionales que son de naturaleza física. Los ejemplos de requerimientos técnicos incluyen la especificación de un paquete de software o de una plataforma de hardware que se requieran. Estos tipos de requerimientos se estudiarán con profundidad en el capítulo 9.

**Formalización de los requerimientos** Generalmente los requerimientos del sistema se documentan de una manera formal para comunicarlos a los involucrados clave del sistema. Este documento sirve como contrato entre los propietarios del sistema y el equipo de desarrollo sobre lo que se va a suministrar en términos de un nuevo sistema. Así, puede someterse a muchas revisiones y repasos antes de que todos concuerden y autoricen el contenido. No hay un nombre o formato estándar para este documento. De hecho, muchas organizaciones usan nombres diferentes tales como declaración de requerimientos, especificación de requerimientos, definición de requerimientos, especificación funcional, y otros, y generalmente el formato se ajusta a las necesidades de la organización. Las compañías que proveen sistemas de información y software al gobierno de los Estados Unidos deben usar las convenciones de formato y terminología especificadas en el documento de estándares publicado por el gobierno MIL-STD-498.<sup>2</sup> Muchas organizaciones han creado sus propios estándares adaptados de MIL-STD-498 debido a que son muy completos y porque muchas personas ya están familiarizadas con ellos. En este libro usaremos el término **documento de definición de requerimientos**, y la figura 5.2 proporciona un ejemplo de uno. Por favor observe que este documento será consolidado con otra información de proyecto para formar la declaración de requerimientos, que es el producto final de la fase de análisis de requerimientos. Un documento de definición de requerimientos deberá consistir de lo siguiente:

---

**documento de definición de requerimientos** Documento formal que comunica los requerimientos de un sistema propuesto a involucrados clave y sirve como un contrato del proyecto de sistemas. Sinónimo de declaración de requerimientos, especificación de requerimientos, y especificación funcional.

<sup>2</sup> MIL-STD-498 es un estándar que fusiona DOD-STD-2167A con DOD-STD-7935A para definir un conjunto de actividades y documentación adecuadas para el desarrollo de los sistemas de armas y de los sistemas automatizados de información.

**FIGURA 5.2**

Muestra de un esbozo de definición de requerimientos

**DOCUMENTO DE DEFINICIÓN DE REQUERIMIENTOS**

1. Introducción
    - 1.1. Propósito
    - 1.2. Fondo
    - 1.3. Alcance
    - 1.4. Definiciones, acrónimos y abreviaturas
    - 1.5. Referencias
  2. Descripción general del proyecto
    - 2.1. Requerimientos funcionales
  3. Requerimientos y restricciones
    - 3.1. Requerimientos funcionales
    - 3.2. Requerimientos no funcionales
  4. Conclusión
    - 4.1. Aspectos relevantes
- Apéndice (opcional)

- Las funciones y los servicios que el sistema debe proveer.
- Los requerimientos no funcionales, incluyendo los aspectos del sistema, las características, y los atributos.
- Las restricciones que limitan el desarrollo del sistema o bajo las cuales debe operar el sistema.
- Información acerca de otros sistemas con las cuales el sistema debe tener una interfase.

¿Quién va a leer el documento de definición de requerimientos? Probablemente este documento es el más leído y más referenciado de toda la documentación de proyecto. Los propietarios y los usuarios del sistema lo usan para especificar sus requerimientos y cambios que puedan surgir. Los gerentes lo usan para preparar los planes y las estimaciones de proyecto, y los desarrolladores para entender lo que se requiere y desarrollar pruebas para validar el sistema. Con esto en mente, es importante observar que los requerimientos se leen más frecuentemente que lo que se escriben. Por lo tanto, el tomarse tiempo para escribirlos correctamente, de manera concisa y clara no sólo va a ahorrar tiempo en términos del calendario, sino que también es más eficiente en costos y reduce el riesgo de errores costosos en los requerimientos. Por lo tanto, realizar la valoración de los requerimientos es un paso necesario para lograr ese objetivo. La validación de los requerimientos se realiza en un borrador final del documento de definición de requerimientos después que se han solicitado todas las entradas por parte de los propietarios y usuarios del sistema. El propósito de esta actividad es que el analista de sistemas se asegure de que los requerimientos se escriban correctamente. Ejemplos de errores que el analista de sistemas podría encontrar son:

- Modelos del sistema que contienen errores.
- Errores tipográficos o gramaticales.
- Conflicto de requerimientos.
- Requerimientos ambiguos o mal redactados.
- Falta de conformación con los estándares de calidad requeridos para el documento.

### > Administración de los requerimientos

Durante la vida útil del proyecto es muy común que surjan nuevos requerimientos y que cambien los existentes después de haber aprobado un documento de definición de requerimientos. Algunos estudios han mostrado que durante la vida del proyecto cambiarán tanto como 50 por ciento o más de los requerimientos antes de que el sistema se ponga en producción. Obviamente que esto puede ser un gran dolor de cabeza para el equipo de

desarrollo. Para ayudar a aliviar los múltiples problemas asociados con los requerimientos que cambian, es necesario realizar una **administración de requerimientos**. Ésta abarca las políticas, los procedimientos, y los procesos que gobiernan el manejo del cambio de un requerimiento. En otras palabras, se especifica cómo debe presentarse una solicitud de cambio, cómo se analiza en cuanto el impacto sobre el alcance, el calendario, y el costo, cómo se aprueba o se rechaza, y cómo se implementa el cambio si se aprueba.

**administración de requerimientos** Proceso de administrar los cambios de los requerimientos.

## Técnicas de exploración

En esta sección presentamos siete técnicas comunes de exploración:

- Muestreo de la documentación, las formas y las bases de datos existentes.
- Investigación y visitas al sitio.
- Observación del ambiente de trabajo.
- Cuestionarios.
- Entrevistas.
- Propuestas de prototipos.
- Planeación conjunta de requerimientos.

Generalmente un analista aplica varias de estas técnicas durante un proyecto de sistemas individual. Para poder seleccionar la técnica más adecuada a usarse en cualquier situación dada, los analistas de sistemas necesitan aprender las ventajas y desventajas de cada una de las técnicas de exploración.

**La ética de la exploración** Durante la exploración, los analistas de sistemas frecuentemente encuentran o analizan información que es de naturaleza sensible. Podría ser un archivo de la estructura de precios de una compañía aeroespacial para una licitación de contrato o aun los perfiles de los empleados, incluyendo salarios, historial de su desempeño, historial médico, y planes de carrera. Los analistas deben tener mucho cuidado para proteger la seguridad y la privacidad de los hechos o datos que se les hayan confiado. Muchas personas y organizaciones en esta atmósfera altamente competitiva están buscando una “ventaja” para sacar ganancia. Los analistas de sistemas descuidados que dejan documentos delicados a la vista sobre sus escritorios, o que discuten en público los datos confidenciales podrían causar un gran daño a la organización o a las personas. Si este tipo de datos cayera en las manos equivocadas, el analista de sistemas podría perder el respeto, la credibilidad, o la confianza de los usuarios y de la gerencia. En algunos casos, el analista sería responsable de la intrusión en la privacidad de una persona y podría resultar legalmente responsable.

La mayoría de las corporaciones hacen todos los esfuerzos para asegurase de que llevan los negocios de una manera ética porque la ley puede exigírselos. Ha habido muchos casos en los que las corporaciones han incurrido en fuertes multas por no llevar apropiadamente los negocios. A este efecto, muchas empresas requieren que sus empleados asistan a seminarios de entrenamiento anuales sobre la ética de la compañía, y refuerzan el aprendizaje exhibiendo banderas o letreros que contienen el código de conducta de la compañía y las declaraciones de ética en el lugar de trabajo en localidades muy visibles. Las políticas de ética de la compañía pueden estar en un formato de fotocopiado que se distribuye a todos los empleados, o en las páginas Web de la compañía, haciéndolas fácilmente accesibles a los empleados sin importar dónde se encuentren actualmente. Las políticas de ética documentan la conducta esperada y requerida. Las violaciones de estas políticas podrían conducir a una acción disciplinaria o aun a un despido. La ética juega un papel crucial en la exploración.

### > Muestreo de la documentación, los formatos y los archivos existentes

Al estudiar un sistema existente, los analistas de sistemas desarrollan una buena percepción del sistema mediante la consulta de la documentación, los formatos y los archivos existentes. Un buen analista siempre sabe obtener hechos de la documentación existente antes que de las personas.

**Recolección de hechos a partir de la documentación existente** ¿Qué tipo de documentos pueden enseñarle algo acerca de un sistema? El primero que el analista desea examinar es el organigrama, el cual sirve para identificar a los propietarios y usuarios individuales clave para un proyecto y sus relaciones de reporte. El analista también desea rastrear el historial que originó el proyecto. Para lograr esto, él deberá recolectar y revisar documentos que describan el problema. Éstos incluyen:

- Memorando interno de la oficina, estudios, minutas, notas en el buzón de sugerencias, quejas de los clientes, y reportes que documenten el área problema.
- Registros contables, revisiones del desempeño, de la medición del trabajo, y otros reportes operativos programados.
- Solicitudes de proyecto de los sistemas de información: pasados y presentes.

Además de los documentos que describen el problema, generalmente hay documentos que describen la función de negocios que está siendo estudiada o diseñada. Estos documentos pueden incluir:

- La declaración de la misión y el plan estratégico de la compañía.
- Los objetivos formales de las unidades de la organización que se están estudiando.
- Los manuales de políticas que pueden imponer restricciones sobre cualquier sistema propuesto.
- Los procedimientos operativos estándar (Standard operating procedures, SOP), las descripciones de puestos, o las instrucciones de tareas para las operaciones cotidianas específicas.
- Formas llenadas que representan las transacciones reales en los diferentes puntos del ciclo de procesamiento.
- Muestras de las bases de datos manuales y computarizadas.
- Muestras de las pantallas y los reportes manuales y computarizados.

También, los analistas frecuentemente revisan la documentación de estudios y diseños del sistema anterior realizados por los analistas y consultores anteriores. Esta documentación puede incluir:

- Diferentes tipos de diagramas de flujo y diagramas.
- Diccionarios o repositorios del proyecto.
- Documentación de diseño, tales como entradas, salidas, y bases de datos.
- Documentación de programa.
- Manuales de operación de las computadoras y manuales de entrenamiento.

Toda la documentación recolectada deberá analizarse para determinar si la información está o no actualizada. La documentación obsoleta no debe descartarse; sin embargo, los analistas deben tener en mente que será necesaria exploración adicional para verificar o actualizar los hechos recolectados. ¿Qué es lo que busca el analista en todo este material? Los elementos que puedan seleccionarse de estos documentos incluyen:

- Los síntomas y (posiblemente) las causas del problema.
- Qué personas en la organización entienden el problema.
- Las funciones de negocios que soportan al sistema presente.
- El tipo de datos que el sistema debe recolectar y reportar.
- Elementos en la documentación que el analista no entiende y que es necesario cubrir en entrevistas.

**Técnicas de muestreo de documentos y archivos** Debido a que sería impráctico estudiar todas las ocurrencias de todos los formatos o registros en un archivo o base de datos, normalmente los analistas de sistemas usan técnicas de **muestreo** para obtener una vista transversal suficientemente amplia para prever lo que puede pasar en el sistema. El analista de sistemas debe tratar de muestrear suficientes formatos que representen la naturaleza y la complejidad total de los datos. Los analistas con experiencia evitan las pifias de muestrear formatos en blanco: éstos dicen poco acerca de cómo se usa el formato en realidad, cuándo no se usa o con qué frecuencia se le da un mal uso. Cuando se estudian documentos o registros de una tabla de base de datos, los analistas deben estudiar suficientes muestras para identificar todas las condiciones y excepciones posibles de procesamiento. Pueden usarse técnicas de muestreo estadístico para determinar si el tamaño de la muestra es suficientemente grande para ser representativa de la población total de los registros o los documentos.

---

**muestreo** Proceso de recolectar una muestra representativa de documentos, formas y registros.

**TABLA 5.3 Tabla parcial de los factores de certeza**

Certeza deseada	Factor de certeza
95%	1.960
90	1.645
80	1.281

Hay muchos factores y aspectos en el muestreo, y ésta es una buena razón para tomar un curso de introducción a la estadística. Una fórmula simple y confiable para determinar el tamaño de la muestra es:

$$\text{Tamaño de la muestra} = 0.25 \times (\text{factor de certidumbre/error aceptable})^2$$

El factor de certidumbre es un valor que simplemente puede buscarse en tablas estadísticas basándose en la certidumbre deseada de que la muestra seleccionada será representativa de la población total. Véase la tabla 5.3 para un ejemplo específico.

Suponga que un analista quiere tener una certeza del 90 por ciento de que una muestra de facturas no va a contener variaciones no muestradas. El tamaño de muestra,  $TM$ , se calcula como sigue:

$$TM = 0.25(1.645/0.10)^2 = 68$$

El analista deberá muestrear 68 facturas para obtener la exactitud deseada. Si se desea un mayor nivel de certeza, se necesita un mayor número de facturas.

Si el analista sabe por experiencia que 1 de cada 10 facturas varía con respecto a la norma, entonces puede reemplazarse el heurístico 0.25 por  $p(1 - p)$  donde  $p$  es la proporción de facturas con variancias:

$$TM = p(1 - p)(1.645/0.10)^2$$

Con el uso de esta fórmula, el analista puede reducir el número de muestras requeridas para obtener la exactitud deseada:

$$TM = 0.10(1 - 0.10)(1.645/0.10)^2 = 25$$

¿Cómo se escogen las 25 facturas? Dos técnicas comúnmente usadas son el muestreo aleatorio y el muestreo estratificado. El **muestreo aleatorio** consiste en seleccionar los datos de la muestra en forma aleatoria o sin importar cómo se seleccionen. Por tanto, simplemente escogemos al azar 25 facturas basándonos en el tamaño de muestra calculado antes. El **muestreo estratificado** es un enfoque bien pensado y sistemático dirigido a reducir la variancia de los datos de la muestra. Para los archivos computarizados, el muestreo estratificado puede realizarse escribiendo un programa sencillo. Por ejemplo, suponga que nuestras facturas están en una base de datos que tiene un volumen de aproximadamente 250 000 facturas. Recuerde que nuestro tamaño de muestra debe incluir 25 facturas. Simplemente escribiremos un programa que imprima un registro de cada 10 000 ( $= 250\,000/25$ ). Para los documentos y archivos manuales, podríamos ejecutar un esquema similar.

---

**muestreo aleatorio** Técnica de muestreo que se caracteriza por no contar con ningún patrón o plan predeterminado para seleccionar los datos de la muestra.

---

**muestreo estratificado** Técnica sistemática de muestreo que intenta reducir la variancia de las estimaciones al dispersar el muestreo (por ejemplo, con una selección de los documentos o los registros mediante una fórmula) y evitar estimaciones muy altas o muy bajas.

## > Investigación y visitas al sitio

Una segunda técnica de exploración es investigar a fondo el dominio del problema. La mayoría de los problemas no son del todo únicos. Otras personas los han resuelto antes que nosotros. Muchas veces las organizaciones contactan o realizan visitas de sitio con compañías que saben que han experimentado antes problemas similares. Si esas compañías tienen “voluntad de compartir”, puede obtenerse información valiosa que puede ahorrar mucho tiempo y costo en el proceso de desarrollo.

Las revistas especializadas en computación y los libros de referencia son una buena fuente de información. Pueden suministrar información sobre cómo otros han resuelto problemas similares. Con los recientes avances en el ciberespacio, los analistas rara vez tienen que dejar sus escritorios para hacer una investigación.

La exploración del Internet y del intranet vía la computadora personal puede suministrar incontables cantidades de información.

Un tipo similar de investigación consiste en visitar a otras compañías o departamentos que han encarado problemas similares. Las membresías en sociedades profesionales tales como la Association for Information Technology Professionals (AITP) o la Association for Information Systems (AIS), entre otras, pueden suministrar una red de contactos útiles.

## > Observación del ambiente de trabajo

**observación** Técnica de exploración en la cual el analista de sistemas participa u observa a una persona que realiza actividades para aprender acerca del sistema.

La observación es una de las técnicas más efectivas de recolección de datos para aprender acerca de un sistema. La **observación** consiste en que el analista de sistemas se convierte en un observador de las personas y de las actividades con objeto de aprender acerca del sistema. Frecuentemente se usa esta técnica cuando se cuestiona la validez de los datos recolectados mediante otros métodos o cuando la complejidad de ciertos aspectos del sistema impide obtener una clara explicación por parte de los usuarios finales.

**Recolección de hechos por observadores en el trabajo** Aun con un plan de observación bien concebido, el analista de sistemas no está seguro de que la observación tenga éxito. La siguiente historia, que aparece en un libro de Gerald M. Weinberg llamado *Rethinking Systems Analysis and Design*, nos da un ejemplo entretenido y al mismo tiempo excelente de algunas de las pifias de la observación.<sup>3</sup>

### La paradoja del ferrocarril

Aproximadamente a 30 millas de Ciudad Gótica se ubica la comunidad que hace viajes de ida y regreso del pueblo de los Suburbios. Cada mañana, miles de pueblerinos tomaban el Ferrocarril Central para trabajar en Ciudad Gótica. Cada tarde, el Ferrocarril Central los regresaba a sus esposas, hijos y perros que los esperaban.

El pueblo de los Suburbios era un suburbio próspero, y a muchas de las esposas les gustaba dejar a los niños y a los perros y pasar la tarde en Ciudad Gótica con sus parejas. Preferían iniciar su tarde de cena y de teatro visitando los fastuosos mercados en Ciudad Gótica. Pero había un problema. Para tener tiempo para ir de compras, una pueblerina tendría que salir de Ciudad Gótica a las 2:30 o 3:00 en la tarde. A esa hora, ningún tren del Ferrocarril Central hacía parada en el Pueblo de los Suburbios.

Algunos pueblerinos observaban que ciertamente un tren Central pasaba por su estación a las 2:30, pero no paraba. Decidieron solicitar al ferrocarril que el tren se programara para parar en el pueblo de los Suburbios. Rápidamente encontraron apoyo en su campaña de puerta en puerta para solicitar votos. Cuando se mandó la solicitud por correo, contenía 253 firmas. Aproximadamente tres semanas después, el comité solicitante recibió la siguiente carta del Ferrocarril Central:

Apreciable comité:

Gracias por su continuo interés en las operaciones del Ferrocarril Central. Nosotros tomamos en serio nuestro cometido de suministrar un servicio que dé respuesta a todas las personas que viven entre nuestras rutas, y apreciamos mucho la retroalimentación en todos los aspectos de nuestro negocio. En respuesta a su solicitud, nuestro representante de servicio al cliente visitó la estación del pueblo de los Suburbios en tres días diferentes, cada vez a las 2:30 de la tarde. Aunque él observó con mucho cuidado, *en ninguna de las tres ocasiones había pasajeros esperando por un tren dirigido al sur*.

Solamente podemos concluir que no hay una demanda real de una parada dirigida al sur a las 2:30, y por lo tanto debemos declinar lamentablemente su solicitud.

Atentamente,

Agente de servicio al cliente

Ferrocarril Central

<sup>3</sup> Gerald M. Weinberg, *Rethinking Systems Analysis and Design*, pp. 23-24. Derechos reservados © 1988, 1982 por Gerald M. Weinberg. Reimpreso con autorización de Dorset House Publishing, 353 W. 12th St., Nueva York, NY 10014 (212-620-4053/800-DH-BOOKS/www.dorsethouse.com). Todos los derechos reservados.

¿Cuáles son las lecciones que se aprenden de la historia anterior? Una es que es necesario usar la técnica de exploración apropiada para el problema a la mano. La observación, en este caso, fue una elección incorrecta. ¿Por qué alguien estaría esperando por el tren de las 2:30 cuando toda la gente del pueblo sabía que el tren no para? Una segunda lección que debe aprenderse es verificar los resultados de la exploración con los usuarios. Basándose en la retroalimentación del usuario, usted puede identificar que es necesario intentar con otras técnicas de exploración para recabar información adicional. Nunca debe saltar a las conclusiones.

**Ventajas y desventajas de la observación** La observación puede ser una técnica de exploración muy útil y benéfica siempre que usted tenga la habilidad para observar todos los aspectos del trabajo que los usuarios están realizando y que el trabajo se esté realizando de la manera acostumbrada. Usted debe estar conciente de los pros y los contras de la técnica de observación. Las ventajas y las desventajas incluyen:

### Ventajas

- Los datos recabados basándose en la observación pueden ser muy confiables. Algunas veces se realizan observaciones para verificar la validez de los datos obtenidos directamente de las personas.
- El analista de sistemas puede ver exactamente lo que se está haciendo. Algunas veces es difícil explicar claramente con palabras las tareas complejas. A través de la observación, el analista de sistemas puede identificar las tareas que se han omitido o que se han descrito con inexactitud por otras técnicas de exploración. También, el analista puede obtener datos que describan el ambiente físico de la tarea (por ejemplo, la disposición física, el tránsito, la iluminación, el nivel de ruido).
- La observación es relativamente barata en comparación con otras técnicas de exploración. Generalmente otras técnicas requieren mayor tiempo de liberación del empleado y más gastos de copiado
- La observación permite que el analista de sistemas haga mediciones del trabajo.

### Desventajas

- Ya que la gente generalmente se siente incómoda cuando está siendo vigilada, inconscientemente puede comportarse de una manera diferente que cuando está siendo observada.
- El trabajo que se esté observando tal vez no incluya el nivel de dificultad o de volumen normalmente experimentado durante ese tiempo.
- Algunas actividades de sistemas pueden tener lugar en horas estrambóticas, causando una inconveniencia de programación para el analista de sistemas.
- Las tareas que se observan están sujetas a diferentes tipos de interrupciones.
- Algunas tareas no siempre serán desempeñadas en la manera en que las observa el analista de sistemas. Por ejemplo, el analista de sistemas pudo haber observado cómo una compañía llenaba varias solicitudes de los clientes. Sin embargo, los procedimientos que el analista de sistemas observó pudieron haber sido los pasos usados para llenar varias solicitudes regulares de los clientes. Si cualquiera de estas solicitudes hubiera sido una solicitud especial (por ejemplo, de bienes que normalmente no se guardan en existencia), el analista de sistemas habría observado un conjunto diferente de procedimientos que se ejecutan.
- Si las personas han estado realizando tareas de una manera que viole los procedimientos estándar de operación, temporalmente pueden desempeñar su trabajo de manera correcta mientras que usted los observa. En otras palabras, la gente puede permitirle ver lo que ellos quieren que usted vea.

**Lineamientos de la observación** ¿Cómo obtiene hechos el analista de sistemas a través de la observación? ¿Se llega simplemente al sitio de observación y se comienza a

registrar todo lo que se ve? No. Antes deberá haber mucha preparación. El analista debe determinar cómo van en verdad a capturarse los datos. ¿Será necesario tener formatos especiales para el registro rápido de los datos? ¿Se molestarán las personas que están siendo observadas porque alguien las observe y registre sus acciones? ¿Cuándo deben observarse los períodos bajos, normales y altos de las operaciones de la tarea? El analista de sistemas debe identificar el momento ideal para observar un aspecto específico del sistema.

Un analista deberá planear observar un sitio cuando hay una carga de trabajo típica. Una vez que se ha observado una carga de trabajo típica, pueden hacerse observaciones durante los períodos pico para recopilar información para medir los efectos causados por el incremento de la carga. Como parte de la observación del analista, deberán obtenerse muestras de documentos o de formatos que usan quienes son observados.

Las técnicas de muestreo que se estudiaron anteriormente también son útiles para la observación. En este caso, la técnica se llama **muestreo del trabajo**; en ella puede realizarse un gran número de observaciones a intervalos aleatorios. Esta técnica es menos incómoda para las personas que están siendo observadas porque el período de observación no es continuo. Al usar el muestreo del trabajo, un analista necesita definir con anticipación las operaciones del trabajo que va a observar. Entonces es necesario calcular el tamaño de la muestra tal como se hizo para el muestreo de documentos y archivos. El analista deberá realizar muchas observaciones aleatorias, teniendo cuidado de observar las actividades en diferentes momentos del día. Contando el número de ocurrencias de cada operación durante las observaciones, el analista se dará cuenta de cómo pasan sus días los empleados.

Los siguientes lineamientos son clave para refinar las habilidades de observación:

- Determine el quién, qué, dónde, cuándo, porqué y cómo de la observación.
- Obtenga el permiso para observar por parte de los supervisores o los gerentes.
- Informe a quienes van a ser observados acerca del propósito de la observación.
- Mantenga un bajo perfil.
- Tome notas durante o inmediatamente después de la observación.
- Revise las notas de la observación con las personas apropiadas.
- No interrumpa a las personas que están trabajando.
- No se centre mucho en actividades triviales.
- No haga suposiciones.

**Viviendo el sistema** En este tipo de observación el analista de sistemas desempeña activamente el papel del usuario por un corto período. Ésta es una de las maneras más efectivas para aprender acerca de los problemas y los requerimientos del sistema. Al “ponerse los zapatos” del usuario, un analista de sistemas obtiene rápidamente una impresión de lo que experimenta el usuario y lo que tiene que hacer para realizar el trabajo. Este tipo de juego del papel le da al analista de sistemas una enseñanza de primera mano de los procesos y las funciones del negocio, así como de los problemas y retos asociados con éstos.

## > Cuestionarios

**cuestionario** Documento que permite al analista recabar información y opiniones de los encuestados.

Otra técnica de exploración es conducir encuestas mediante **cuestionarios**. El cuestionario puede producirse en gran cantidad y distribuirse a los encuestados, quienes entonces pueden llenar el cuestionario cuando tengan tiempo. Los cuestionarios le permiten al analista recolectar hechos de un gran número de personas al tiempo que se mantienen respuestas uniformes. Al enfrentarse a una audiencia grande, ninguna otra técnica de exploración puede tabular los mismos hechos con la misma eficacia.

**Recolección de hechos mediante el uso de cuestionarios** Frecuentemente los analistas de sistemas han criticado el uso de los cuestionarios. Muchos analistas de sistemas objetan que las respuestas carecen de información confiable y útil. Sin embargo, los cuestionarios pueden ser un medio efectivo para recopilar los hechos, y muchas de estas críticas pueden atribuirse al uso inapropiado o poco efectivo de los cuestionarios. Antes de

usar los cuestionarios, un analista deberá entender los pros y los contras asociados con su uso:

### Ventajas

- La mayoría de los cuestionarios pueden responderse rápidamente. La gente puede completar y devolver los cuestionarios con toda comodidad.
- Los cuestionarios son un medio relativamente barato de recopilar datos provenientes de un gran número de personas.
- Los cuestionarios permiten que las personas mantengan el anonimato. Por lo tanto, es más probable que las personas suministren los hechos reales en vez de decirle lo que piensan que su jefe querría que ellos hicieran.
- Las respuestas pueden tabularse y analizarse rápidamente.

### Desventajas

- Con frecuencia el número de encuestados es bajo.
- No existe garantía de que una persona responda o se explaye a todas las preguntas.
- Los cuestionarios tienden a ser inflexibles. No hay oportunidad de que el analista de sistemas obtenga información voluntaria o parafrasee las preguntas que pudieron haber sido mal interpretadas.
- No es posible que el analista de sistemas observe y analice el lenguaje corporal del encuestado.
- No hay una oportunidad inmediata para aclarar una respuesta vaga o incompleta a cualquier pregunta.
- Los buenos cuestionarios son difíciles de preparar.

**Tipos de cuestionarios** Hay dos formatos para los cuestionarios: el formato libre y el formato fijo. Los **cuestionarios de formato libre** se diseñan para permitir a los usuarios que ejerciten más libertad o flexibilidad en sus respuestas a cada pregunta.

Aquí hay dos ejemplos de las preguntas con formato libre:

- ¿Qué reportes recibe actualmente y cómo los usa?
- ¿Hay algún problema con estos reportes (por ejemplo, son inexactos, la información es insuficiente, o son difíciles de leer, usar o ambas cosas)? Si es así, por favor explique.

**cuestionario de formato libre** Cuestionario diseñado para ofrecer al encuestado más laxitud en la respuesta. Se formula una pregunta, y el encuestado registra la respuesta en el espacio provisto después de la pregunta.

Es obvio que las respuestas a estas preguntas pueden ser difíciles de tabular. También es posible que las respuestas de quienes responden no concuerden con las preguntas formuladas. Con objeto de asegurar respuestas útiles en los cuestionarios de formato libre, el analista deberá expresar las preguntas en oraciones simples y no usar palabras —tales como *bueno*— que puedan interpretarse en forma distinta por diferentes encuestados. El analista también deberá formular preguntas que puedan responderse con tres o menos oraciones. De otra manera, el cuestionario puede consumir más tiempo que el que el encuestado desea dedicar.

El segundo tipo de cuestionario es el de formato fijo. Los **cuestionarios de formato fijo** son más rígidos ya que requieren que el usuario seleccione una respuesta de un conjunto de respuestas posibles previamente definido. Dada cualquier pregunta, el encuestado debe escoger de las respuestas disponibles. Esto hace que los resultados sean mucho más fáciles de tabular. Por otro lado, el encuestado no puede suministrar información adicional que pudiera resultar valiosa.

Hay tres tipos de preguntas de formato fijo:

**cuestionario de formato fijo** Cuestionario que contiene preguntas que requieren la selección de una respuesta entre respuestas disponibles predefinidas.

1. Para las *preguntas de opción múltiple*, se le dan al encuestado varias respuestas de las cuales escoger. Deberá decirse al encuestado si puede seleccionarse más de una respuesta. Algunas preguntas de opción múltiple permiten respuestas muy breves de formato libre cuando no son aplicables ninguna de las respuestas estándar. Los ejemplos de las preguntas de formato fijo de opción múltiple son:

¿Piensa usted que las órdenes atrasadas ocurren con demasiada frecuencia?  
 SÍ       NO

¿Es útil el reporte actual de cuentas por cobrar que usted recibe?  
 SÍ       NO

Si la respuesta es no, por favor explique.

2. Para las *preguntas de calificación*, se le da al encuestado un enunciado y se le pide que use las respuestas suministradas para emitir una opinión. Para evitar el provocar un sesgo, deberá haber un número igual de calificaciones positivas y negativas. El siguiente es un ejemplo de una pregunta de calificación con formato fijo:

La implementación de los descuentos en las cantidades causaría un aumento en las órdenes de los clientes.

- Definitivamente de acuerdo
- De acuerdo
- Sin opinión
- En desacuerdo
- Definitivamente en desacuerdo

3. Para las *preguntas de jerarquización*, se le dan al encuestado varias respuestas posibles, que deben jerarquizarse en orden de preferencia o de experiencia. Un ejemplo de una pregunta de jerarquización con formato fijo es:

Jerarquice las siguientes transacciones de acuerdo con el tiempo que usted ocupa en procesarlas:

- \_\_\_\_\_ órdenes de los clientes nuevos
- \_\_\_\_\_ cancelaciones de órdenes
- \_\_\_\_\_ órdenes modificadas
- \_\_\_\_\_ pagos

**Cómo desarrollar un cuestionario** Los buenos cuestionarios pueden ser difíciles de desarrollar. El siguiente procedimiento puede ser útil para el desarrollo de una encuesta efectiva:

1. Determine qué hechos y opiniones deben recolectarse y de quién debe obtenerlas. Si el número de personas es grande, considere el uso de un grupo de encuestados más pequeño seleccionado aleatoriamente.
2. Basándose en los hechos y en las opiniones buscadas, determine si las preguntas de formato libre o fijo darán las mejores respuestas. Frecuentemente se usa un formato combinado que permite la aclaración opcional de formato libre de las respuestas de formato fijo.
3. Escriba las preguntas. Examínelas en cuanto a errores de construcción y posibles malas interpretaciones. Asegúrese de que las preguntas no revelen su sesgo personal o sus opiniones. Revise las preguntas.
4. Ensaye las preguntas en una pequeña muestra de encuestados. Si sus encuestados tuvieran problemas con éstas o si las respuestas no fueran útiles, revise las preguntas.
5. Duplique y distribuya el cuestionario.

## > Entrevistas

---

**entrevista** Técnica de exploración mediante la cual el analista de sistemas recolecta información de las personas a través de la interacción cara a cara.

Generalmente se reconoce que la entrevista personal es la técnica de exploración más importante y de uso más frecuente. Las **entrevistas** personales consisten en preguntar los requerimientos a través de una interacción directa cara a cara. Las entrevistas pueden usarse para alcanzar alguno o todos los objetivos siguientes: indagar hechos, verificar hechos, aclarar hechos, generar entusiasmo, hacer que se involucre el usuario final, identificar los requerimientos, y solicitar ideas y opiniones. Hay dos papeles que se asumen en una entrevista. El analista de sistemas es el *entrevistador*, responsable de la organización y la conducción de la entrevista. El usuario del sistema o el propietario del sistema es el *entrevistado*, a quien se le pide que responda a una serie de preguntas.

Puede haber uno o más entrevistadores, entrevistados o ambos. En otras palabras, las entrevistas pueden conducirse de uno a uno o de muchos a muchos. Desafortunadamente, muchos analistas de sistemas son malos entrevistadores. En esta sección usted aprenderá a conducir entrevistas adecuadas.

**Recolección de hechos por los usuarios que hacen la entrevista** Las personas son el elemento más importante de un sistema de información. Más que cualquier otra cosa, la gente quiere estar en cosas activas. Ninguna otra técnica de exploración pone tanto énfasis en la gente como las entrevistas. Pero las personas tienen valores, prioridades, opiniones, motivaciones, y personalidades diferentes. Por lo tanto, para usar la técnica de la entrevista, un analista de sistemas debe poseer buenas habilidades de relaciones humanas para tratar con efectividad a diferentes tipos de personas. Al igual que con otras técnicas de exploración, la entrevista no es el mejor método para todas las situaciones; tiene sus ventajas y sus desventajas, que deben sopesarse con las de otras técnicas de exploración para cada situación de exploración:

### Ventajas

- Las entrevistas dan al analista una oportunidad para motivar al entrevistado para que responda libre y abiertamente a las preguntas. Al establecer una armonía, el analista de sistemas puede darle al entrevistado una percepción de que está contribuyendo activamente al proyecto de sistemas.
- Las entrevistas permiten que el analista de sistemas intente obtener más retroalimentación del entrevistado.
- Las entrevistas permiten que el analista de sistemas adapte o parafrasee las preguntas para cada persona.
- Las entrevistas le dan al analista una oportunidad para observar la comunicación no oral del entrevistado. Un buen analista de sistemas puede ser capaz de obtener información al observar los movimientos corporales y las expresiones faciales del entrevistado así como al escuchar las respuestas orales a las preguntas.

### Desventajas

- La entrevista es un enfoque de exploración que consume mucho tiempo, y por tanto es muy costosa.
- El éxito de las entrevistas depende mucho de las habilidades en relaciones humanas del analista de sistemas.
- Una entrevista puede ser impráctica debido a la ubicación del entrevistado.

---

**entrevista no estructurada** Entrevista que se conduce solamente con un objetivo o tema general en mente y con pocas preguntas específicas, si es que las hay. El entrevistador cuenta con el entrevistado para proveer un marco y dirigir la conversación.

---

**entrevista estructurada** Entrevista en la cual el entrevistador tiene un conjunto específico de preguntas para hacérselas al entrevistado.

---

**pregunta de respuesta abierta** Pregunta que permite al entrevistado responder de cualquier manera que parezca apropiada.

---

**pregunta de respuesta cerrada** Pregunta que restringe las respuestas ya sea a selecciones específicas o a respuestas cortas y directas.

**Tipos y técnicas de la entrevista** Hay dos tipos de entrevistas, la no estructurada y la estructurada. Las **entrevistas no estructuradas** se caracterizan porque contienen preguntas generales que permiten al entrevistado dirigir la conversación. Frecuentemente, este tipo de entrevista se sale del camino, y el analista debe estar preparado para redirigir la entrevista hacia el principal objetivo o tema. Por esta razón, las entrevistas no estructuradas generalmente no funcionan bien para el análisis y el diseño de sistemas. Las **entrevistas estructuradas** consisten en que el entrevistador formula preguntas específicas diseñadas para obtener información específica de parte del entrevistado. Dependiendo de las respuestas del entrevistado, el entrevistador dirigirá preguntas adicionales para obtener una aclaración o una ampliación. Algunas de estas preguntas pueden ser planeadas y otras, espontáneas.

Las entrevistas no estructuradas tienden a que se formulen **preguntas de respuesta abierta**. Este tipo de preguntas da a los entrevistados una gran libertad en sus respuestas. Un ejemplo de una pregunta de extremo abierto es: “¿Por qué no está satisfecho con el reporte de cuentas incobrables?” Las entrevistas estructuradas tienden a que se formulen más **preguntas de respuesta cerrada** que están diseñadas para suscitar respuestas cortas y directas de parte del entrevistado. Ejemplos de preguntas así son: “¿Está recibiendo el reporte de las cuentas incobrables a tiempo?” y “¿Contiene el reporte de cuentas incobrables información exacta?” Desde un punto de vista realista, la mayoría de las preguntas se sitúan entre los dos extremos.

## > Cómo conducir una entrevista

El éxito de un analista de sistemas depende, al menos parcialmente, de la capacidad para entrevistar. Una entrevista de éxito incluye la selección de las personas apropiadas para la entrevista, la preparación extensa para la entrevista, la conducción apropiada de la entrevista, y el seguimiento de la misma. Aquí examinamos cada uno de estos pasos con más detalle. Supongamos que el analista ha identificado la necesidad de una entrevista y ha determinado exactamente qué tipos de hechos y opiniones se necesitan.

**Selección de los entrevistados** El analista de sistemas debe entrevistar a los usuarios finales del sistema de información que se está estudiando. Un organigrama formal ayudará a identificar a estas personas y sus responsabilidades. Antes de la entrevista, el analista debe intentar aprender tanto como sea posible acerca de cada persona (sus fortalezas, sus temores, sus prejuicios y sus motivaciones) entonces la entrevista puede armarse para que considere las características de la persona.

El analista debe hacer una cita con el entrevistado y nunca simplemente aparecerse. La cita debe limitarse a un lapso de entre media hora y una hora. Entre mayor sea el nivel de dirección del entrevistado, deberá programarse menos tiempo. Si el entrevistado es un trabajador de oficina, de servicio o un obrero, el analista debe obtener el permiso del supervisor antes de programar la entrevista. También es importante asegurarse de que el lugar de la entrevista esté disponible durante el tiempo programado. Las entrevistas nunca deben conducirse en presencia de los compañeros de oficina del analista o de los colegas del entrevistado.

**Preparación para la entrevista** La preparación es la clave de una entrevista exitosa. Un entrevistado puede detectar fácilmente cuando un entrevistador no está preparado y puede resentir la falta de preparación porque se desperdicia tiempo valioso. Cuando se hace la cita, el entrevistado deberá ser notificado acerca del tema de la entrevista. Para asegurarse de que se cubran todos los aspectos pertinentes del tema, el analista debe preparar un *guión de entrevista*, el cual es una lista de verificación de preguntas específicas que el entrevistador le hará al entrevistado. El guión de entrevista también puede contener preguntas de seguimiento que se formularán solamente si las respuestas a otras preguntas justifican las respuestas adicionales. En la figura 5.3 se presenta una muestra de un guión de entrevista. Observe que la agenda se prepara cuidadosamente con el tiempo específico asignado a cada pregunta. También debe apartarse tiempo para formular preguntas de seguimiento y para redirigir la entrevista. Las preguntas deben seleccionarse y expresarse con cuidado. La mayoría de las preguntas comienzan con el tipo acostumbrado de formulación de quién, qué, cuándo, dónde, por qué y cuánto. Deben evitarse los siguientes tipos de preguntas:

- *Preguntas cargadas*, tales como “¿Tenemos que tener estas dos columnas en el reporte?” La pregunta transmite la opinión personal del entrevistado sobre el tema.
- *Preguntas con intención*, tales como “Usted no va a usar este CÓDIGO DE OPERADOR, ¿verdad?” La pregunta conduce a que el entrevistado responda, “No, por supuesto que no”, independientemente de su opinión real.
- *Preguntas sesgadas*, tales como “¿Cuántos códigos necesitamos para la CLASIFICACIÓN DE ALIMENTOS en el ARCHIVO DE INVENTARIOS? Pienso que 20 deberían de cubrirlo.” Estos tipos de preguntas sesgadas van a influir en el entrevistado.

Los entrevistadores siempre deben evitar preguntas amenazantes o críticas. El propósito de la entrevista es investigar, no evaluar ni criticar. Lineamientos adicionales para las preguntas incluyen:

- Use un lenguaje claro y conciso.
- No incluya su opinión como parte de la pregunta.
- Evite preguntas largas o complejas.
- Evite las preguntas amenazantes.
- No use “usted” cuando se refiera a un grupo de personas.

Entrevistado:	Jeff Bentley, Gerente de cuentas por cobrar	
Fecha:	19 de enero de 2003	
Hora:	1:30 p. m.	
Lugar:	Sala 223, Edificio de administración	
Tema:	Política actual de investigación de crédito	
Tiempo asignado	Pregunta u objetivo del administrador	Respuesta del entrevistado
1 a 2 min.	<p><b>Objetivo</b>  Comienza la entrevista:  <ul style="list-style-type: none"> <li>• Nos presentamos</li> <li>• Gracias Sr. Bentley por su valioso tiempo</li> <li>• Enunciar el propósito de la entrevista: obtener una comprensión de las políticas existentes de investigación de crédito.</li> </ul> </p>	
5 min.	<p><b>Pregunta 1</b>  ¿Qué condiciones determinan si se aprueba una solicitud de crédito del cliente?  <b>Seguimiento</b></p>	
5 min.	<p><b>Pregunta 2</b>  ¿Cuáles son las posibles decisiones o acciones que podrían tomarse una vez que estas condiciones han sido evaluadas?  <b>Seguimiento</b></p>	
3 min.	<p><b>Pregunta 3</b>  ¿Cómo se notifica a los clientes cuando no se aprueba su solicitud de crédito?  <b>Seguimiento</b></p>	
1 min.	<p><b>Pregunta 4</b>  Después que se aprueba una nueva solicitud de crédito y se coloca en el archivo que contiene las solicitudes que pueden llenarse, un cliente puede pedir que se haga una modificación a la solicitud. ¿Tendría que pasar ésta nuevamente por la aprobación de crédito si el costo total de la nueva solicitud sobrepasa al costo original?  <b>Seguimiento</b></p>	
1 min.	<p><b>Pregunta 5</b>  ¿Quiénes son las personas que realizan las investigaciones de crédito?  <b>Seguimiento</b></p>	
1 a 3 min.	<p><b>Pregunta 6</b>  ¿Puedo obtener el permiso para hablar con estas personas para aprender específicamente cómo llevan a cabo el proceso de investigación de crédito?  <b>Seguimiento</b>  Si así es: ¿Cuál sería el momento apropiado para reunirme con cada uno de ellos?</p>	
1 min.	<p><b>Objetivo</b>  Termino de la entrevista:  <ul style="list-style-type: none"> <li>• Agradezca al Sr. Bentley por su cooperación y asegúrele que estará recibiendo una copia de lo que se obtuvo durante la entrevista.</li> </ul> </p>	
21 minutos	Tiempo asignado para preguntas y objetivos	
9 minutos	Tiempo asignado para preguntas de seguimiento y redirección	
30 minutos	Tiempo asignado para la entrevista (1:30 p.m. a 2:00 p.m.)	
<b>Comentarios generales y notas:</b>		

**FIGURA 5.3** Muestra de un guión para entrevista

**Conducción de la entrevista** Respete el tiempo de su entrevistado. Arréglese para hacer la entrevista. Generalmente esto significa que usted se arreglará de manera diferente para entrevistar a los gerentes que para entrevistar a los trabajadores en el muelle de carga. Si la entrevista se celebra en una sala de reuniones en vez de la oficina del entrevistado, llegue temprano para asegurarse de que esté acondicionada apropiadamente.

Inicie la entrevista agradeciendo al entrevistado. Enuncie el propósito y la duración de la entrevista y cómo van a usarse los datos recabados. Entonces monitoree el tiempo de modo que cumpla su promesa.

Formule preguntas de seguimiento. Intente hasta que entienda los requerimientos del sistema. Pregunte especialmente acerca de las condiciones de excepción. Formule preguntas del tipo ¿qué pasa si?, tales como “¿Qué sucede si el cheque no pasa?” o “¿Qué pasa si un producto no está en existencia?”

Escuche con atención, observe al entrevistado, y tome notas acerca de sus respuestas verbales y no verbales. Es muy importante mantener la entrevista en su curso; esto implica anticipar la necesidad de adaptar la entrevista, si es necesario. Frecuentemente las preguntas pueden ser omitidas si han sido respondidas anteriormente o pueden eliminarse si se determina que son irrelevantes, basándose en respuestas anteriores.

Aquí hay un conjunto de reglas que debe seguir un entrevistador:

Haga	Evite
<ul style="list-style-type: none"> <li>• Vístase adecuadamente.</li> <li>• Sea cortés.</li> <li>• Escuche cuidadosamente.</li> <li>• Mantenga el control de la entrevista.</li> <li>• Explore.</li> <li>• Observe los gestos y la comunicación no oral.</li> <li>• Sea paciente.</li> <li>• Mantenga al entrevistado en calma.</li> <li>• Mantenga su autocontrol.</li> <li>• Termine a tiempo.</li> </ul>	<ul style="list-style-type: none"> <li>• Suponer que una respuesta esté terminada o que no lleva a ningún lado.</li> <li>• Revelar pistas orales y no orales.</li> <li>• Usar la jerga.</li> <li>• Revelar sus sesgos personales.</li> <li>• Hablar en lugar de escuchar.</li> <li>• Suponer cualquier cosa acerca del tema o el entrevistado.</li> <li>• Uso de la grabadora: una señal de habilidades deficientes para escuchar.</li> </ul>

Concluya la entrevista expresando su aprecio y proveyendo respuestas a las preguntas interpuestas por el entrevistado. La conclusión es muy importante para mantener la armonía y la confianza con el entrevistado.

**Seguimiento de la entrevista** Para ayudar a mantener una buena armonía y confianza con los entrevistados, el entrevistador deberá mandarles un memorando que resuma la entrevista. Este documento deberá recordarles a los entrevistados sus contribuciones al proyecto de sistemas y permitirles la oportunidad de aclarar malas interpretaciones en que el entrevistador haya incurrido durante la entrevista. Además, a los entrevistados deberá dárseles la oportunidad de ofrecer información adicional que hayan omitido durante la entrevista.

**Cómo escuchar** Cuando la mayoría de la gente habla acerca de las habilidades de comunicación, piensan en hablar y escribir. Rara vez se menciona la habilidad de escuchar, pero puede ser la más importante durante el proceso de la entrevista. Con objeto de conducir una entrevista exitosa, el entrevistador debe diferenciar entre oír y escuchar: “Oír es reconocer que alguien está hablando, escuchar es entender lo que el orador quiere comunicar”.<sup>4</sup>

En realidad hemos estado condicionados durante toda la vida a no escuchar. Considere por ejemplo, como podemos ignorar a nuestros hermanos y hermanas que pelean mientras disfrutamos de nuestro CD favorito, cómo aprendemos a estudiar bloqueando nuestras distracciones tales como compañeros de cuarto ruidosos. Hemos aprendido a no escuchar, pero también podemos aprender cómo escuchar en forma efectiva y productiva.

<sup>4</sup> Thomas R. Gildersleeve, *Successful Data Processing Systems Analysis* (Englewood Cliffs, NJ: Prentice Hall, 1978), p. 93.

Al trabajar con usuarios que tratan de resolver sus problemas, los analistas pueden encontrar que hacer que los usuarios se comuniquen puede ser difícil. Los siguientes lineamientos pueden ayudar a abrir los canales de comunicación.

- *Llegue a la sesión con una actitud positiva.* El entrevistador debe aprovechar cualquier situación de la mejor manera, y verla como una experiencia divertida y placentera.
- *Haga que la otra persona se tranquilice.* El presentar una actitud agradable y animada puede ayudar a que la persona se relaje. El entrevistador debe comenzar hablando acerca de los intereses o las aficiones de la persona. Algunas veces el mostrar un interés en la vida personal del entrevistado puede servir para romper el hielo y hacer que la persona se relaje más.
- *Deje que la otra persona se entere que usted está escuchando.* El entrevistador siempre deberá mantener el contacto visual cuando escuche y asentir con la cabeza o emitir un “ája” para reconocer lo que la persona está diciendo. Una buena postura y el inclinarse hacia delante le dirá al orador que el entrevistador está realmente interesado en lo que la persona está diciendo.
- *Haga preguntas.* El entrevistador deberá formular preguntas para asegurarse de que entiende claramente lo que la persona está diciendo o para aclarar un punto. Esto mostrará que el entrevistador está escuchando; también le dará a la otra persona la oportunidad de ampliar la respuesta.
- *No suponga nada.* Una de las peores cosas que puede hacer un entrevistador es actuar como si tuviera prisa. Por ejemplo, si un entrevistador supone lo que la otra persona va a decir y la interrumpe y termina la oración, posiblemente pierda lo que la persona trataba de decir e irrite al entrevistado. O si éste es interrumpido porque el entrevistador ya ha oído la información y piensa que no es aplicable al tema de la entrevista, podría perderse una pieza valiosa de información. ¡No suponga nada! El comentarista de la televisión Art Linkletter aprendió esta lección en su popular show de la televisión, *Kids Say the Darnedest Things*, cuando le preguntó a un niño una cuestión filosófica:

En mi show una vez hice que un niño me dijera que él quería ser un piloto de aerolínea. Le pregunté qué haría si todos los motores se pararan en medio del Océano Pacífico. Él dijo: “Primero les diría a todos que se aprieten los cinturones de seguridad, y luego buscaría mi paracaídas y saltaría”.

Mientras que el auditorio se destornillaba de risa, centré mi atención en el joven para ver si se estaba pasando de listo. Las lágrimas que brotaban de sus ojos me alertaron de su pesar más que cualquier cosa que pudiera haber dicho, así es que le pregunté por qué dijo tal cosa. Su respuesta reveló la sólida lógica de un niño: “Voy a conseguir gasolina...Voy a regresar!”<sup>5</sup>

- *Tome notas.* El proceso de tomar notas sirve para dos propósitos. Primero, al grabatear notas breves mientras que la otra persona está hablando, usted le da la impresión de que lo que dice es suficientemente importante como para ser escrito. Segundo, las notas ayudan al entrevistador a recordar más tarde los puntos principales de la reunión.

**El lenguaje corporal y la proxemía** ¿Qué es el lenguaje de cuerpo, y por qué debería preocuparse un analista de sistemas acerca de esto durante el proceso de la entrevista? El **lenguaje corporal** es toda la información no verbal que una persona está comunicando. El lenguaje de cuerpo es una forma de comunicación no verbal que todos usamos y de la cual generalmente no estamos conscientes.

---

**lenguaje corporal** Información no verbal que comunicamos.

Los estudios han determinado un hecho desconcertante: del total de los sentimientos de una persona, sólo el 7 por ciento se comunica verbalmente (con palabras), mientras que el 38 por ciento se comunica por el tono de voz empleado y el 55 por ciento por las expresiones faciales y del cuerpo. Si usted solamente escucha las palabras de alguien, está perdiendo la mayor parte de lo que la persona tiene que decir.

<sup>5</sup> Donald Walton, *Are You Communicating? You Can't Manage without It.* (Nueva York: McGraw-Hill, 1989), p. 31.

Para esta discusión nos centraremos solamente en tres aspectos del lenguaje corporal: la expresión facial, el contacto visual, y la postura. La *expresión facial* implica que algunas veces usted puede entender cómo se siente una persona observando las expresiones en su rostro. Muchas emociones comunes son expresiones faciales fácilmente reconocibles asociadas con éstas. Sin embargo, la cara es una de las partes más controladas del cuerpo. Algunas personas que se dan cuenta de que sus expresiones con frecuencia revelan lo que están pensando, son muy buenas para disimularlas.

Otra forma de comunicación no verbal es el *contacto visual*. Éste es el aspecto menos controlado de la expresión facial. ¿Alguna vez ha hablado con alguien que no lo mire directamente? ¿Qué fue lo que sintió? Una falta continua del contacto visual puede indicar incertidumbre. Una mirada normal en general dura de tres a cinco segundos; sin embargo, el tiempo de contacto visual directo debe aumentar con la distancia. Es necesario que los analistas tengan cuidado de no usar un contacto visual excesivo con los usuarios que parecen sentirse amenazados de modo que no los intimiden más. El contacto visual directo puede causar sentimientos fuertes, ya sean positivos o negativos, en otras personas.

La *postura* es el aspecto menos controlado del cuerpo. Como tal, la postura corporal contiene mucha información para el analista astuto. Los miembros de un grupo que están de acuerdo tienden a exhibir la misma postura. Un buen analista vigilará los cambios de postura del auditorio que pudieran indicar ansiedad, desacuerdo, o aburrimiento. Normalmente un analista deberá mantener una posición corporal “abierta”, transmitiendo el acercamiento, la aceptación y la receptividad. En circunstancias especiales, el analista puede escoger usar un ángulo de confrontación con la cabeza hacia el frente o en un ángulo de 90 grados con respecto a otra persona con objeto de establecer control y dominación.

Además de la información comunicada por el lenguaje corporal, las personas también se comunican vía la proxemia. La **proxemia**, la relación entre las personas y el espacio alrededor de ellas, es un factor de comunicaciones que el analista experimentado puede controlar.

La gente todavía tiende a ser muy territorial con su espacio. Observe dónde se sientan sus compañeros de clase en alguno de sus cursos que no tienen pupitres asignados. O la próxima vez que usted participe en una conversación con alguien, muévase deliberadamente mucho más cerca o más lejos de la persona y vea lo que pasa. Un buen analista reconoce cuatro zonas espaciales:

- *Zona íntima*: menos de 1.5 pies.
- *Zona personal*: de 1.5 pies a 4 pies.
- *Zona social*: de 4 pies a 12 pies.
- *Zona pública*: más de 12 pies.

Ciertos tipos de comunicaciones tienen lugar solamente en alguna de estas zonas. Por ejemplo, un analista conduce la mayoría de las entrevistas con los usuarios del sistema en la zona personal. Pero puede ser necesario que el analista regrese a la zona social si el usuario muestra cualquier signo (lenguaje corporal) de estar incómodo. Algunas veces el contacto visual creciente puede compensar una distancia larga que no puede modificarse. Muchas personas usan las fronteras de la zona social como una distancia de “respeto”.

Hemos examinado algunas de las maneras informales en las que la gente comunica sus sentimientos y sus reacciones. Un buen analista usará toda la información disponible, no sólo la comunicación escrita o verbal de otros.

## > Elaboración de prototipos de identificación

Otro tipo de técnica de exploración es la elaboración de prototipos, la cual se introdujo en el capítulo 3 para usarse en el desarrollo rápido de aplicación (rapid application development, RAD). Como recordará, el concepto detrás de la elaboración de prototipos es la construcción de un pequeño modelo de trabajo de los requerimientos del usuario o un diseño propuesto para un sistema de información. Este tipo de elaboración de prototipo en general es una técnica de diseño, pero el enfoque puede aplicarse anteriormente al

**proxemia** Relación entre las personas y el espacio a su alrededor.

ciclo de vida de desarrollo del sistema para realizar el análisis de exploración y de requerimientos. El proceso de construcción de un prototipo para el propósito de identificar los requerimientos se denomina la **elaboración del prototipo de identificación**.

Frecuentemente se aplica la elaboración de prototipos de identificación a los proyectos de desarrollo de sistemas, en especial en los casos donde el equipo de desarrollo está teniendo problemas para definir los requerimientos del sistema. La filosofía es que los usuarios van a reconocer sus requerimientos cuando los vean. Es importante que el prototipo se desarrolle rápidamente de modo que pueda usarse durante el proceso de desarrollo. En general, se construyen prototipos sólo en las áreas donde los requerimientos no se entienden claramente. Esto implica que puede excluirse mucha funcionalidad deseable e ignorarse el aseguramiento de la calidad. También, los requerimientos no funcionales tales como el desempeño y la confiabilidad pueden ser menos restrictivos que lo que sería para el producto final. Con frecuencia se usan tecnologías diferentes de las que se emplean para el software final en la construcción de los prototipos de identificación. En estos casos, lo más probable es que los prototipos se descarten cuando se termine el sistema. Este enfoque de “desechar” se usa principalmente para recabar información y desarrollar ideas para el concepto de sistema. Tal vez no se entiendan con claridad muchas áreas de un sistema propuesto, o algunos aspectos pueden ser un desafío técnico para los desarrolladores. La creación de prototipos de identificación permite a los desarrolladores así como a los usuarios entender mejor y refinar los aspectos implicados en el desarrollo del sistema. Esta técnica ayuda a minimizar el riesgo de entregar un sistema que no satisfaga las necesidades de los usuarios o que no pueda cumplir los requerimientos técnicos.

La elaboración de prototipos de identificación tiene sus ventajas y desventajas, lo que debe sopesarse con aquellas de otras técnicas de exploración para cada situación de exploración:

### elaboración del prototipo de identificación

Acto de construir un representante a pequeña escala o modelo de trabajo de los requerimientos de los usuarios con objeto de identificar o verificar esos requerimientos.

#### Ventajas

- Permite que los usuarios y los desarrolladores experimenten con el software y desarrollen una comprensión de cómo podría trabajar el sistema.
- Ayuda a determinar la factibilidad y la utilidad del sistema antes de incurrir en altos costos de desarrollo.
- Sirve como un mecanismo de entrenamiento para los usuarios.
- Ayuda a construir los planes y escenarios de prueba del sistema para usarse al último en el proceso de pruebas del sistema.
- Puede minimizar el tiempo invertido en la exploración y ayudar a definir requerimientos más estables y confiables.

#### Desventajas

- Puede ser necesario entrenar a los desarrolladores en el enfoque de elaboración de prototipos.
- Los usuarios pueden desarrollar expectativas poco realistas basándose en el desempeño, la confiabilidad, y las características del prototipo. Los prototipos solamente pueden simular la funcionalidad del sistema y su naturaleza es incompleta. Debe tenerse cuidado en educar a los usuarios acerca de este hecho para no desorientarlos.
- La elaboración de prototipos puede prolongar el programa de desarrollo y aumentar los costos de desarrollo.

## > Planeación conjunta de requerimientos

Muchas organizaciones están usando la sesión grupal de trabajo como un sustituto de entrevistas separadas y numerosas. Un ejemplo del enfoque de la sesión grupal de trabajo es la **planeación conjunta de requerimientos (joint requirements planning, JRP)**, en la cual se conducen reuniones de grupo altamente estructuradas con el objeto de identificar y analizar problemas y definir los requerimientos del sistema. Éstas y otras técnicas similares en general requieren de un extenso entrenamiento para trabajar como está pensado. Sin embargo, pueden reducir significativamente el tiempo invertido en la exploración en una o más fases del ciclo de vida. La JRP se está haciendo cada vez más común en la planeación de sistemas y en el análisis de sistemas para obtener un consenso del grupo sobre los problemas, objetivos, y requerimientos. En esta sección, usted aprenderá acerca

### planeación conjunta de requerimientos (JRP)

Proceso mediante el cual se conducen reuniones de grupo altamente estructuradas con el propósito de analizar problemas y definir requerimientos.

de los participantes en una sesión JRP y sus papeles. También estudiaremos qué hacer con la planeación y conducción de una sesión JRP, las herramientas y las técnicas que se usan durante una sesión de JRP, y los beneficios que se logran a través de JRP.

**Los participantes en JRP** Las sesiones de planeación conjunta de requerimientos incluyen una amplia variedad de participantes y de papeles. Se espera que cada participante asista y participe activamente en la sesión completa de JRP. Examinemos los diferentes tipos de personas que participan en una sesión típica de JRP y sus papeles:

- **Patrocinador:** Cualquier sesión exitosa de JRP requiere que una sola persona, llamada *patrocinador*, sirva como su campeón. Normalmente esta persona es un individuo que está en la dirección (*no* en la administración TI o IS) y que tiene autoridad sobre los diferentes departamentos y usuarios que van a participar en el proyecto de sistemas. El patrocinador da todo su apoyo al proyecto de sistemas al alentar a los usuarios designados a que participen en forma activa y por su propia voluntad en la sesión de JRP. Recordando el enfoque del “compromiso progresivo” al desarrollo de sistemas, es el patrocinador (propietario del sistema) quien toma las decisiones finales en relación con la dirección de avance o no avance del proyecto.

El patrocinador juega un papel visible durante una sesión de JRP al dar inicio a la junta introduciendo a los participantes. Frecuentemente, el patrocinador también hará comentarios finales sobre la sesión y trabaja íntimamente con el líder del JRP para planear la sesión al ayudar a identificar a las personas provenientes de la comunidad de usuarios que deberán asistir y al determinar la fecha y la ubicación de la sesión de JRP.

- **Facilitador:** Las sesiones de JRP incluyen a una persona que juega el papel de líder o facilitador. Generalmente, el *facilitador de JRP* es responsable de conducir todas las sesiones que se celebren para un proyecto de sistemas. Esta persona es alguien que tiene excelentes habilidades de comunicación, posee la capacidad para negociar y resolver conflictos de grupo, tiene un buen conocimiento del negocio, buenas habilidades para la organización, es imparcial con las decisiones que se van a enfrentar, y no reporta a ninguno de los participantes de la sesión de JRP.

Algunas veces es difícil encontrar una persona dentro de la compañía que posea todas estas cualidades. Entonces, con frecuencia las compañías deben suministrar un extenso entrenamiento para JRP o contratar un experto externo a la organización para cumplir con este papel. Muchos analistas de sistemas están entrenados para ser facilitadores de JRP.

El papel del facilitador de JRP es planear la sesión de JRP, conducir la sesión, y dar seguimiento a los resultados. Durante la sesión, el facilitador es responsable de encabezar la discusión, alentando a los asistentes para que participen activamente, resolviendo los conflictos clave que puedan surgir, y asegurándose de que se alcancen las metas y los objetivos de la reunión. La responsabilidad del facilitador de JRP es establecer las reglas de campo que se seguirán durante la reunión y asegurarse de que los participantes se adhieran a estas reglas.

- **Usuarios y gerentes:** La planeación conjunta de requerimientos incluye a varios participantes provenientes de los sectores de usuarios y gerencial de una organización a quienes se les concede tiempo de comisión de sus actividades cotidianas para dedicarse a una participación activa en las sesiones de JRP. Estos participantes son seleccionados por el patrocinador, quien debe ser cuidadoso para asegurarse de que cada persona tenga el conocimiento de los negocios para contribuir durante las sesiones de exploración. El patrocinador del proyecto debe ejercer su autoridad y estímulo para asegurarse de que estas personas se dedicarán a una participación activa.

Una sesión típica de JRP puede incluir cualquier número de personas desde uno relativamente pequeño de usuarios/gerentes hasta una docena o más. El papel de los usuarios durante una sesión de JRP es comunicar con efectividad las reglas y los requerimientos de negocios, revisar los prototipos de diseño, y tomar decisiones aceptables. El papel de los gerentes durante una sesión de JRP es aprobar los objetivos del proyecto, establecer las prioridades del mismo, aprobar los calendarios y los costos, y aprobar las necesidades de entrenamiento ya identificadas y los planes de

implementación. En general, se confía en que tanto los usuarios como los gerentes van a asegurarse que sus factores críticos de éxito están siendo considerados.

- *Secretario(s)*: Una sesión de JRP también incluye uno o más *secretarios*, quienes son responsables de llevar el registro relativo a todo lo que se discuta en la reunión. Estos registros se publican y se distribuyen a los asistentes inmediatamente después de la reunión con objeto de conservar el impulso que ha sido establecido por la sesión de JRP y sus miembros. La necesidad de publicar rápidamente los registros se refleja por el hecho de que los secretarios están usando cada vez más las herramientas de CASE para capturar muchos hechos (datos documentados para el usuario y modelos de proceso) que se comunican durante una sesión de JRP. Entonces, es conveniente que los secretarios posean un profundo conocimiento del análisis y diseño de sistemas y que sean hábiles en el uso de las herramientas de CASE. Los analistas de sistemas frecuentemente juegan este papel.
- *Equipo de TI*: Una sesión de JRP también puede incluir varias personas de TI, quienes principalmente escuchan y toman notas en relación con los aspectos y los requerimientos voceados por los usuarios y los gerentes. Normalmente, el personal de TI no hace comentarios a menos que se le invite. En vez de ello, las preguntas o inquietudes que tengan en general son canalizadas al facilitador de JRP inmediatamente antes o después de la sesión de JRP. Es el facilitador de JRP quien inicia y facilita la discusión de los aspectos por los usuarios y los gerentes.

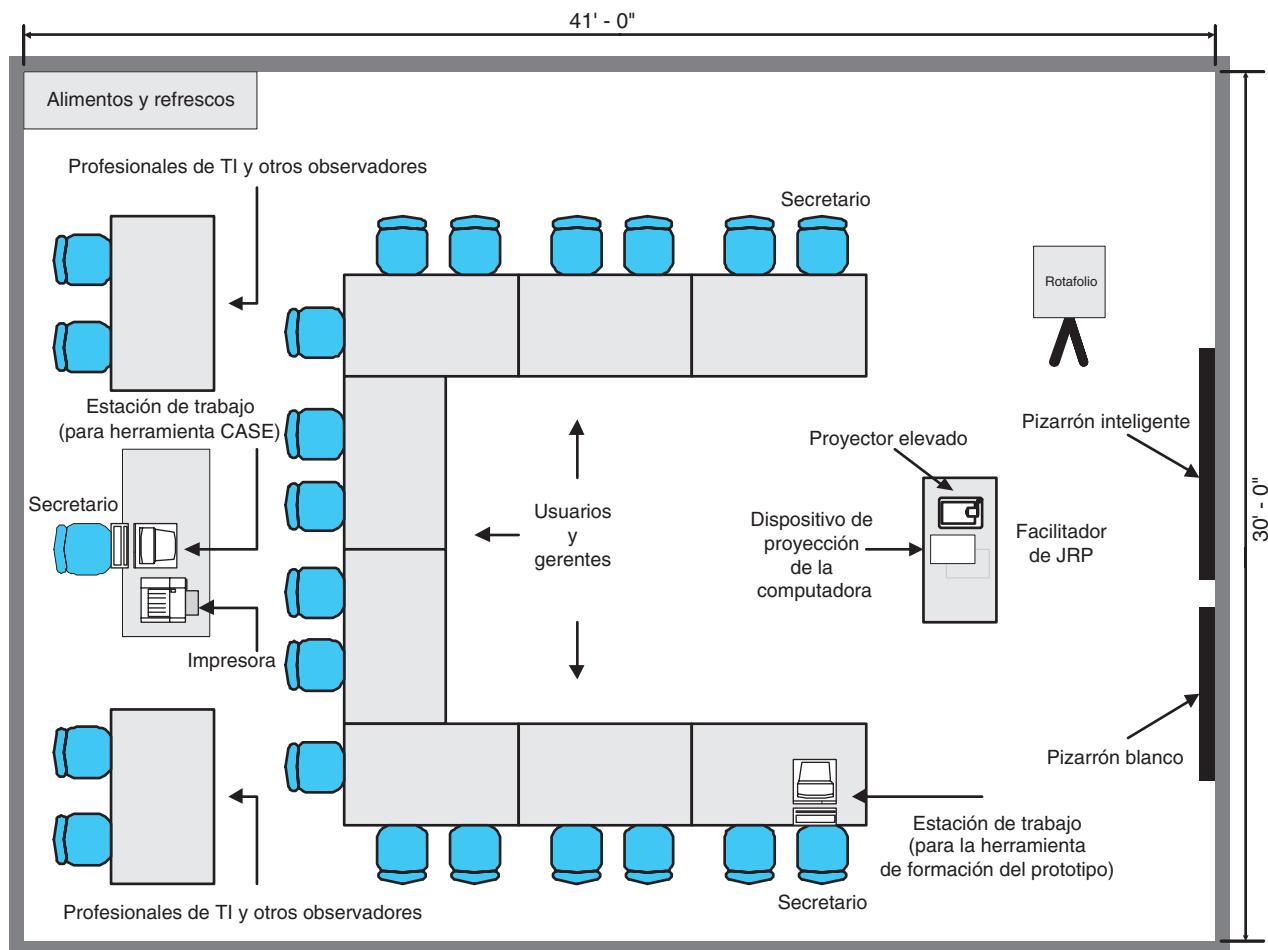
El equipo de TI en la sesión de JRP generalmente incluye miembros del equipo de proyecto, quienes pueden trabajar personalmente con el secretario para desarrollar modelos y otra documentación relacionada con los hechos comunicados durante la reunión. También puede llamarse a especialistas para que adquieran información con respecto a los aspectos técnicos e inquietudes especiales que puedan surgir. Si la situación lo justifica, el facilitador de JRP puede urgir a un profesional de TI para que encare el aspecto técnico.

**Cómo planear las sesiones de JRP** La mayoría de las sesiones de JRP abarcan de tres a cinco días y ocasionalmente duran hasta dos semanas. El éxito de cualquier sesión de JRP depende de la planeación apropiada y del desempeño efectivo del plan. Es necesaria alguna preparación mucho antes de realizar la sesión de JRP. Antes de planear una sesión de JRP, el analista debe trabajar íntimamente con el patrocinador ejecutivo para determinar el alcance del proyecto que se va a abordar a través de las sesiones de JRP. También es importante que se determinen los requerimientos de alto nivel y las expectativas de cada sesión de JRP. En general esto implica entrevistar a personas seleccionadas que son responsables de los departamentos o de las funciones que deben ser abordadas por el proyecto de sistemas. Finalmente, antes de planear la sesión de JRP, el analista debe asegurarse de que el patrocinador ejecutivo tenga deseos de dedicar gente, tiempo, y otros recursos a la sesión.

La planeación de una sesión de JRP implica tres pasos: selección de una ubicación para la sesión de JRP, selección de los participantes en JRP, y la preparación de una agenda que deberá seguirse durante la sesión de JRP. Examinemos cada uno de estos pasos de planeación en detalle:

1. *Selección de una ubicación para las sesiones de JRP*: Siempre que sea posible, las sesiones de JRP deberán conducirse lejos del lugar de trabajo de la compañía. La mayoría de los hoteles o las universidades locales tienen instalaciones diseñadas para atender las reuniones de grupo. Al celebrarse la sesión de JRP en una ubicación extra muros, los asistentes pueden concentrarse en los aspectos y actividades relacionados con la sesión de JRP y evitar interrupciones y distracciones que ocurrirían en su lugar de trabajo habitual. Independientemente de la ubicación de la sesión de JRP, deberá requerirse que todos los participantes asistan y deberá prohibírseles que regresen a su lugar de trabajo habitual.

Es común que una sesión de JRP requiera de varias salas. Se necesita una sala de conferencias en la cual pueda reunirse el grupo completo para abordar los aspectos de JRP. También, si la sesión de JRP incluye mucha gente, pueden ser necesarias varias salas más pequeñas para que se reúnan grupos separados de personas y que se adentren en la discusión de aspectos específicos.



**FIGURA 5.4** Disposición típica de una sala para sesiones de JRP

La sala de conferencias o sala principal de reuniones deberá alojar cómodamente a todos los asistentes. Deberá estar totalmente equipada con mesas, sillas y otros elementos que satisfagan las necesidades de todos los asistentes. La figura 5.4 ilustra una disposición típica de una sala para una sesión de JRP. Las ayudas visuales típicas para una sala de JRP deberán incluir un pizarrón blanco, un pizarrón inteligente o un pizarrón negro; uno o más rotafolios; y uno o más proyectores.

La sala también deberá incluir el equipo de cómputo necesario para los secretarios para registrar hechos y aspectos comunicados durante la sesión. La computadora deberá incluir paquetes de software para apoyar los diferentes tipos de registros o documentación que se vaya a capturar y publicar posteriormente por los secretarios. Este software puede incluir la herramienta CASE, un procesador de palabra, una hoja de cálculo, un paquete de presentación, un software para la elaboración de prototipos, una impresora, una copiadora (o un acceso rápido), y capacidad de proyección de cómputo. Como un lineamiento, el equipo de cómputo (excepto el que se use para la formación del prototipo) deberá ubicarse en la parte de atrás de la sala de modo que no interfiera o se convierta en una distracción para los participantes en el JRP. El foco de la sesión deberá ser la interacción personal de los participantes y no la tecnología.

Finalmente, la sala deberá estar equipada para teleconferencias de modo que puedan participar usuarios de localidades distantes; deberá incluir cuadernos de notas y lápices para los usuarios, los gerentes y otros asistentes. A estos últimos también deberá proporcionárseles etiquetas, tarjetas de ubicación, bocadillos, y bebidas para

que estén tan cómodos como sea posible. Las comodidades para los participantes son muy importantes ya que las sesiones de JRP son muy intensas y frecuentemente duran todo el día.

2. *Selección de los participantes de JRP:* Como se mencionó anteriormente, los participantes seleccionados incluyen al facilitador de JRP, el (los) secretario(s), y los representantes de la comunidad de usuarios. Los usuarios deberán ser personas clave que dominen el área de negocios. Desafortunadamente, con frecuencia los gerentes son muy dependientes de estas personas para llevar las áreas de negocios y se resisten a liberarlos de sus tareas. Así, el analista debe asegurarse de que la gerencia está comprometida con el proyecto de JRP y que quieren no solamente permitir sino también requieren que estas personas clave participen.

También pueden seleccionarse diferentes profesionales de TI para que participen en la sesión de JRP. Generalmente todas las personas de TI asignadas al equipo de proyecto participan en la sesión de JRP. También pueden seleccionarse otros especialistas de TI para abordar aspectos técnicos específicos al proyecto.

3. *Cómo preparar la agenda para una sesión de JRP:* La preparación es la clave para una sesión exitosa de JRP. El facilitador de JRP debe preparar documentación para informar brevemente a los participantes acerca del alcance y los objetivos de las sesiones. Además, deberá prepararse una agenda para cada sesión de JRP y distribuirse antes de cada sesión. La agenda determina los aspectos que se van a discutir durante la sesión y el tiempo asignado a cada tema.

La agenda deberá contener tres partes: la introducción, el cuerpo y la conclusión. La introducción pretende comunicar las expectativas de la sesión, las reglas de campo, e influir o motivar a los asistentes para participar. El cuerpo pretende detallar los temas o aspectos que van a abordarse en la sesión de JRP. Finalmente, la conclusión representa el tiempo apartado para resumir la sesión del día y para recordar a los asistentes de los aspectos no solucionados (para desarrollarse posteriormente).

**Cómo conducir una sesión de JRP** La sesión de JRP comienza con comentarios iniciales, presentaciones, y un breve panorama de la agenda y de los objetivos de la sesión. El facilitador de JRP dirigirá la sesión siguiendo el guión preparado. Para conducir la sesión con éxito, como facilitador deberá seguir estos lineamientos:

- No se desvíe de la agenda en forma irrazonable.
- Manténgase dentro del programa (a los temas de la agenda se les asignan tiempos específicos).
- Asegúrese de que el secretario puede tomar notas (esto puede implicar que los usuarios y los gerentes vuelvan a enunciar sus comentarios más lentamente o con más claridad).
- Evite el uso de jerga técnica.
- Aplique habilidades para resolver conflictos.
- Permita recesos largos.
- Aliente el consenso de grupo.
- Aliente la participación del usuario y de la gerencia sin permitir que las personas dominen la sesión.
- Asegúrese de que los asistentes se adhieran a las reglas de campo establecidas para la sesión.

Uno de los objetivos de una sesión de JRP es generar posibles ideas para resolver un problema. Un enfoque es la lluvia de ideas. La **lluvia de ideas** consiste en alentar a los participantes para generar tantas ideas como sea posible, sin analizar la validez de las mismas.

La lluvia de ideas es una técnica formal que requiere disciplina. Deberán usarse los siguientes lineamientos para asegurar una lluvia de ideas efectiva:

1. Aíslle a las personas apropiadas en un lugar que esté libre de distracciones e interrupciones.
2. Asegúrese de que todos entienden el propósito de la junta (generar ideas para resolver el problema) y céntrese en el (los) problema(s).

---

**Iluvia de ideas** Técnica para generar ideas al alentar a los participantes para que ofrezcan tantas ideas como sea posible en un corto tiempo sin ningún análisis hasta que se hayan agotado todas las ideas.

3. Asigne a una persona para que registre las ideas. Esta persona deberá usar un rotafolios, un pizarrón o un proyector elevado que pueda ver todo el grupo.
4. Recuérdelle a todos las reglas de la lluvia de ideas:
  - a) Ser espontáneo. Enunciar las ideas tan pronto como ocurran.
  - b) No se permite ninguna crítica, análisis, o evaluación de ninguna clase mientras que las ideas se están generando. Cualquier idea puede ser útil, aunque solamente sea para originar otra.
  - c) La meta es la cantidad de ideas, no necesariamente la calidad.
5. Dentro de un tiempo específico, los miembros del equipo expresan sus ideas tan rápido como puedan pensarlas.
6. Despues que el grupo haya agotado sus ideas y todas hayan sido registradas, entonces, y solamente entonces, es que éstas pueden ser analizadas y evaluadas.
7. Refine, combine y amplifique las ideas que se generaron anteriormente.

Con un poco de práctica y atención a estas reglas, la lluvia de ideas puede ser una técnica muy efectiva para generar ideas para resolver problemas.

Como se mencionó anteriormente, el éxito de una sesión de JRP es muy dependiente de la planeación y de las habilidades del facilitador de JRP y de los secretarios. Estas habilidades mejoran a través del entrenamiento apropiado y la experiencia. Por lo tanto, las sesiones de JRP en general concluyen con un cuestionario de evaluación para que lo llenen los participantes. Las respuestas ayudarán a asegurar la posibilidad de los futuros éxitos de JRP.

El producto final de una sesión de JRP es comúnmente un documento formal escrito. En general este documento es creado por el facilitador de JRP y por los secretarios. Es esencial para confirmar las especificaciones acordadas durante la sesión por todos los participantes. Es obvio que el contenido y la organización de las especificaciones dependen de los objetivos de la sesión de JRP. El analista puede proveer un conjunto distinto de especificaciones a los diferentes participantes basándose en su papel: por ejemplo, un gerente puede recibir más que una versión resumida del documento suministrado a los usuarios participantes (especialmente en los casos en que los propietarios del sistema tengan una participación mínima real en la sesión de JRP).

**Los beneficios de JRP** La planeación conjunta de requerimientos ofrece muchos beneficios como un enfoque alternativo de exploración y desarrollo. Más y más compañías están comenzando a vislumbrar sus ventajas e incorporando el JRP en sus metodologías existentes. Una sesión de JRP conducida con efectividad ofrece los siguientes beneficios:

- JRP involucra activamente a los usuarios y a la gerencia en el proyecto de desarrollo (alentándolos a “apropiarse” del proyecto).
- JRP reduce el tiempo requerido para desarrollar sistemas. Esto se logra reemplazando las entrevistas tradicionales de uno a uno que consumen mucho tiempo de cada usuario y cada gerente con las reuniones de grupo. Estas últimas permiten obtener más fácilmente un consenso entre los usuarios y los gerentes, así como resolver la información y los requerimientos conflictivos.
- Cuando el JRP incorpora la elaboración de prototipos como un medio para confirmar los requerimientos y obtener las aprobaciones de diseño, se vislumbran los beneficios de la elaboración de prototipos.

El lograr una sesión exitosa de JRP depende del facilitador de JRP y de su habilidad para planear y facilitar la sesión de JRP.

## Una estrategia de exploración

Un analista necesita un método organizado para recabar hechos. Los analistas sin experiencia frecuentemente inician de inmediato las entrevistas. Ellos piensan, “Ve a la gente. ¡Ahí es donde están los hechos reales!” Este enfoque no reconoce un hecho importante de la vida: Las personas deben terminar su trabajo cotidiano. Usted podría pensar, “Pero yo pensaba que usted ha estado diciendo que el sistema es para personas y que la partici-

pación directa del usuario final en el desarrollo de sistemas es esencial. ¿No están ustedes contradiciéndose?

No del todo. El tiempo es oro. Desperdiciar el tiempo de los usuarios finales es aprovechar el dinero de su compañía. Para aprovechar bien el tiempo transcurrido con usuarios finales, los analistas no deberían entrar directamente en las entrevistas. Los analistas primero deberían recolectar todos los hechos que puedan usando otros métodos. Considere la siguiente estrategia paso por paso:

1. Aprenda de documentos existentes, formas, informes, y archivos. Los analistas pueden aprender bastante sin tener contacto con las personas.
2. Si es apropiado, observe el sistema en acción.
3. Dados todos los hechos ya recabados, diseñe y distribuya cuestionarios para aclarar las cosas que no se entienden completamente.
4. Conduzca entrevistas (o las sesiones de trabajo en grupo). Como la mayor parte de los hechos pertinentes ya ha sido recolectada por los métodos de contacto de usuario bajo, las entrevistas pueden usarse para verificar y aclarar los asuntos y los problemas más difíciles. (Alternativamente, considere usar las técnicas JRP para reemplazar o complementar entrevistas.)
5. (Optativo.) Construya prototipos de identificación para cualquier requerimiento funcional que no sean comprendidos o para los requerimientos que necesitan ser validados.
6. Haga un seguimiento. Use técnicas de exploración apropiadas para verificar hechos (usualmente las entrevistas o la observación).

La estrategia no es sagrada. Aunque una estrategia de exploración debería ser desarrollada para cada fase pertinente de desarrollo de sistemas, cada proyecto es único. Algunas veces la observación y los cuestionarios pueden ser inadecuados. Pero la idea siempre debería ser colecciónar tantos hechos como sea posible antes de usar las entrevistas.

Este capítulo le inició en una gran variedad de técnicas para identificar los requerimientos de un sistema de información. La mayoría de metodologías de desarrollo de sistemas requieren algún nivel de documentación y el análisis de los requerimientos del sistema. A continuación, los capítulos restantes de esta parte presentan varias herramientas y técnicas de la documentación de sistemas que pueden usarse durante la fase de análisis del desarrollo de sistemas. La mayoría de ustedes proseguirán directamente al capítulo 6, “Modelado de requerimientos del sistema con los casos de uso”. Los modelos de casos de uso sirven como fundamento para el desarrollo de modelos subsiguientes para modelar los requerimientos adicionales de sistemas y se presentan en los capítulos 7 a 9.

## Resumen

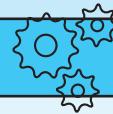


1. El proceso y las técnicas que un analista de sistemas usa para identificar, analizar, y entender los requerimientos de sistema son llamados identificación de requerimientos.
2. Los requerimientos de sistema especifican lo que el sistema de información debe hacer, o qué propiedad o calidad debe tener el sistema.
3. El proceso de identificación de requerimientos consta de las siguientes actividades:
  - a) El análisis e identificación del problema.
  - b) La identificación de requerimientos.
  - c) La documentación y el análisis de los requerimientos.
  - d) La administración de los requerimientos.
4. La exploración de los hechos es una técnica que se usó a través del ciclo de desarrollo completo, pero es sumamente crítica en la fase de análisis de requerimientos.
5. Una herramienta popular usada por los equipos de desarrollo para identificar, analizar, y solucionar problemas es el diagrama Ishikawa.
6. La conducción de negocios de una manera ética es una práctica requerida, y los analistas necesitan ser más conscientes de las implicaciones de no ser éticos.
7. Hay siete técnicas de exploración comunes:
  - a) El muestreo de documentos y archivos existentes puede proveer muchos hechos y detalles sin que sea necesaria poca o ninguna comunicación personal directa. El analista deberá recolectar documentos históricos, manuales y formas de operaciones en los negocios, y documentos de sistemas de información.
  - b) La investigación es una técnica a menudo pasada por alto basada en el estudio de otras aplicaciones similares. Ahora se ha hecho más conveniente con la Internet y la Web (WWW). Las visitas del sitio Web son una forma especial de investigación.
  - c) La observación es una técnica de exploración en la cual el analista estudia a la gente haciendo su trabajo.
  - d) Se usan cuestionarios para recolectar hechos similares de un gran número de individuos.
  - e) Las entrevistas son lo más popular pero son la técnica de exploración de hechos que consume más tiempo. Al entrevistar, el analista se encuentra individualmente con las personas para recabar información.
    - i) Cuando la mayoría de la gente habla de habilidades de comunicación, piensan acerca de hablar y escribir. La de escuchar apenas se menciona, pero puede ser lo más importante, especialmente durante el proceso de entrevista.
- ii) Los estudios de investigación han determinado un hecho sorprendente: Del total de los sentimientos de una persona, sólo siete por ciento se comunica oralmente (con palabras), mientras que 38 por ciento se comunica por el tono de voz usado y 55 por ciento por las expresiones facial y corporal. Si usted sólo escucha las palabras de alguien, pierde la mayor parte de lo que la persona tiene que decir. Los analistas de sistemas experimentados ponen mucha atención al lenguaje corporal y a la proxemía.
- f) La elaboración de prototipos de identificación se aplica frecuentemente a los proyectos de desarrollo de sistemas, en especial en casos donde el equipo de desarrollo tiene problemas para definir los requerimientos del sistema. La filosofía es que los usuarios reconocerán sus requerimientos cuando los vean. Es importante que el prototipo sea desarrollado con rapidez a fin de que pueda ser usado durante el proceso de desarrollo.
- g) Muchos analistas encuentran fallas en las entrevistas: las entrevistas por separado frecuentemente conducen a hechos, opiniones, y prioridades conflictivos. El resultado final es un gran número de entrevistas de seguimiento, reuniones grupales o ambas. Por esta razón, muchas organizaciones están usando una sesión de trabajo grupal conocida como sesión de planeación conjunta de requerimientos como un sustituto de la entrevista.
  - i) Las sesiones de planeación conjunta de requerimientos incluyen una gran variedad de participantes y los papeles. Se espera que cada participante asista y participe activamente durante toda la sesión JRP.
  - ii) Una sesión efectiva JRP incluye una planeación extensa. Planificar para una sesión JRP involucra tres pasos: la selección de un lugar para la sesión JRP, la selección de participantes en la JRP, y la preparación de un orden del día para ser seguido durante la sesión JRP.
8. Para ayudar a aliviar los muchos problemas asociados con el cambio de requerimientos, hay que realizar la administración de requerimientos. Ésta abarca las políticas, los procedimientos, y los procesos que gobiernan cómo se maneja un cambio de un requerimiento.
9. Porque “el tiempo es oro”, es sabio y práctico para el analista de sistemas usar una estrategia de exploración para maximizar el valor del tiempo ocupado con los usuarios finales.
  - a) Aprenda de documentos, formas, informes, y archivos existentes. Los analistas pueden aprender bastante sin tener contacto con las personas.

- b) Si es apropiado, observe el sistema en actividad.
  - c) Dados todos los hechos ya recabados, diseñe y distribuya cuestionarios para aclarar las cosas que no son completamente comprendidas.
  - d) Conduzca entrevistas (o las sesiones de trabajo grupal). Como la mayor parte de los hechos pertinentes ya ha sido recolectada por los métodos de contacto de usuario bajo, las entrevistas pueden usarse para verificar y aclarar los asuntos y los problemas más difíciles. (Alternativamente,
- considere usar las técnicas JRP para reemplazar o complementar entrevistas.)
- e) (Optativo.) Construya prototipos de identificación para los requerimientos funcionales que no se comprendan o para los requerimientos que necesitan ser validados.
  - f) Haga un seguimiento. Use técnicas de exploración apropiadas para verificar hechos (usualmente las entrevistas o la observación.)

## Preguntas de repaso

1. ¿Cuál es la importancia de conducir el proceso de identificación de requerimientos?
2. ¿Cuáles son las consecuencias posibles si usted fracasa al identificar correcta y completamente los requerimientos de sistema?
3. ¿Cuáles son algunos de los criterios considerados como críticos en la definición de los requerimientos de sistema?
4. ¿De qué actividades consta el proceso de identificación de requerimientos?
5. Describa brevemente el propósito y los componentes de un diagrama Ishikawa.
6. ¿Qué técnica se usa comúnmente en la fase de identificación de requerimientos? ¿Por qué es importante eso?
7. ¿Por qué es esencial el análisis de requerimientos?
8. Al recolectar los hechos de la documentación existente, ¿qué clase de documentos debería revisar el analista de sistema?
9. ¿Cuáles son algunas de las desventajas de recolectar hechos observando a los empleados en su ambiente de trabajo? ¿Cómo pueden manejar estas desventajas los analistas de sistemas?
10. ¿Cuáles son los tipos de cuestionarios para encuesta que los analistas de sistemas pueden utilizar para recopilar la información y las opiniones?
11. ¿Cuáles son algunas de las formas con las que usted puede ayudar a abrir las líneas de comunicación en una entrevista?
12. ¿Qué es la planeación conjunta de requerimientos (JRP)?
13. ¿Por qué la JRP se ha hecho popular?
14. ¿Por qué es tan importante el facilitador de JRP?
15. ¿Cuál es la preocupación principal al seleccionar un sitio para sesiones de JRP?



## Problemas y ejercicios

1. Usted maneja un proyecto que se pospuso dos veces porque su financiamiento fue desviado a proyectos de prioridad más alta. Los dueños del sistema no quieren que eso ocurra otra vez, así es que están llenos de ansiedad por iniciar el nuevo sistema y hacer que se construya tan rápido como sea posible. A usted lo presionan mucho para que no ocupe más de un par de días en la identificación de los requerimientos. Si hace falta cualquier cosa, le dicen, puede componerse más tarde. Usted realmente quiere complacerlos, pero se hace presente una pequeña voz de cautela. ¿Cuáles son las consecuencias y los costos potenciales de recorrer a la carrera el proceso de identificación de los requerimientos?
  2. Usted ha aprendido la importancia de asegurarse de que los requerimientos están correctamente identificados. Si no, ¿cómo sabe usted cuándo tiene un requerimiento correcto; es decir, qué criterios
- debe cumplir cada requerimiento para ser considerado correcto?
3. ¿Qué error común comete a menudo un analista de sistemas nuevo al analizar un problema? ¿Cuáles son las consecuencias potenciales de este error? ¿Qué herramienta puede usarse para ayudar a evitar este problema?
  4. Los desarrolladores de sistemas usan técnicas de exploración en cada fase de proyecto. ¿Es la exploración más importante durante la fase de análisis de requerimientos que para otras fases? ¿Por qué sí o por qué no?
  5. ¿Qué asuntos éticos podrían surgir durante el proceso de exploración, y cómo deberían manejarse?
  6. ¿Cuáles son algunas de las técnicas y herramientas comunes que un analista de sistemas puede usar para documentar los hallazgos iniciales? ¿Debería esperar el analista de sistemas que los requerimientos sean cabales y correctos en este punto? En caso

- de que no, ¿cuáles son los problemas comunes? ¿Cuál debe ser el enfoque del equipo de proyecto en este punto?
7. ¿Cuál es el producto que se crea una vez que el análisis de requerimientos se termina? ¿Por qué se necesita este producto y qué es lo que incluye? ¿Quiénes son la audiencia, los usuarios (o ambos) de este producto, y por qué razones?
  8. Usted es un analista de sistemas en una compañía de desarrollo del software que ha sido contratada para hacer la fase de análisis de requerimientos para una organización grande. ¿Cuáles son las tres categorías de documentación existente que usted debería recolectar durante la identificación de los requerimientos? ¿Cuáles son algunos ejemplos de cada uno de estos tres tipos de documentación? ¿Con qué debería tener cuidado el analista de sistemas en la recolección de la documentación?
  9. Suponga que usted es un analista de sistemas en un proyecto que involucra modificar el proceso de orden de ventas. Como su compañía recibe aproximadamente 2 500 órdenes de ventas al día, ¿cuántas necesita usted muestrear si quiere la certeza de 95 por ciento de que tiene cubiertas todas las variacio-

nes? ¿Qué ocurre si el número de órdenes de ventas por día fuera de 25 000?

10. Las encuestas y los cuestionarios se usan frecuentemente para recolectar hechos. ¿Cuáles son las ventajas y las desventajas de los cuestionarios? ¿Cuándo escogería los cuestionarios de formato libre en vez de los cuestionarios del formato fijo? ¿Cuál es un método para determinar la efectividad de un cuestionario?
11. ¿Cuáles son algunas de las razones para usar la planeación conjunta de requerimientos (JRP) como una técnica de exploración? ¿Cuál debería ser la base para seleccionar cuáles usuarios y gerentes participarán en la sesión JRP, y generalmente quién los selecciona? ¿Qué habilidades deberían poseer el facilitador y el secretario? ¿Cuál es el papel del equipo de TI (tecnología de la información) durante las sesiones de JRP? ¿Cuál es la duración típica de las sesiones de JRP?
12. Provea al menos cinco de los factores críticos de éxito para las sesiones JRP.
13. ¿Qué es lo que *no* debería hacer un analista al empezar la etapa de exploración de la identificación de requerimientos, sin importar cuán tentador sea?

## Proyectos e investigación



1. Los analistas de sistemas deben tener pericia en el análisis de problemas. Cuando los analistas de sistemas comienzan, a menudo encuentran difícil diferenciar los síntomas de los problemas, e identificar las causas reales del problema. Una herramienta que puede ayudar a los analistas a aprender a hacer esto es el diagrama Ishikawa, o de espina de pescado.
  - a) Encuentre y seleccione un problema que su empresa, escuela, u otra organización actualmente trata de resolver. Describa este problema.
  - b) Siga el proceso descrito en este capítulo y cree un diagrama Ishikawa.
  - c) ¿Con cuáles categorías comenzó usted en el diagrama, y que cuáles añadió durante el proceso?
  - d) ¿Coadyuvó este diagrama en encontrar la(s) causa(s) real(es) del problema? ¿La(s) causa(s) que usted originalmente pensó resultó(aron) ser verdadera(s), o algo diferente?
2. Observar el ambiente de trabajo es una técnica que antecede a la era de la informática, pero que todavía puede ser altamente efectivo. ¡Aunque no es aplicable para cada situación, el observar qué hacen las personas realmente y cómo lo hacen puede ser en algunos casos mucho más exacto que preguntarles! Seleccione un sistema (ya sea hipotético o verdadero) y haga lo siguiente:
  - a) Provea una visión general del sistema y qué trata usted de aprender del sistema para un proyecto.
  - b) Desarrolle un plan de observación de trabajo usando los lineamientos de este capítulo. El formato depende de usted, pero generalmente no debería necesitar más que 1 a 2 páginas.
  - c) Desarrolle un plan de muestreo del trabajo, y describa los procedimientos de muestreo que usted usará.
  - d) ¿Qué piensa acerca de este método comparado con otros métodos de exploración?
3. Usted es un analista de sistemas que trabaja en un proyecto para desarrollar una intranet para una organización grande con muchos miles de empleados trabajando en oficinas a todo lo largo de los Estados Unidos. Ésta será la primera intranet de la organización, y la gerencia ejecutiva la quiere para ayudar a aumentar eficiencia de los empleados y el compromiso con la empresa. Como parte de la exploración, es necesario recopilar información de los empleados con respecto al contenido de la intranet y la funcionalidad. Debido al tamaño y la distribución geográfica de la organización, así como las restricciones de tiempo del proyecto, el tiempo y los recursos son insuficientes para las entrevistas personales, así es que usted ha decidido que es necesario un cuestionario.
  - a) ¿Cuáles son los hechos y las opiniones que usted necesita recolectar?
  - b) ¿Deberá encuestarse a todos los empleados de la organización? ¿Por qué sí o por qué no? En

- caso de que no todos los empleados deban ser encuestados, ¿cómo seleccionaría usted a los que deban ser encuestados?
- c) ¿Qué formato piensa usted que funcionaría mejor para este cuestionario de encuesta? ¿Si fuera el formato fijo, qué tipo(s) de preguntas del formato fijo debería usar?
  - d) ¿Qué longitud debería tener el cuestionario de encuesta para obtener la información necesaria sin desanimar a los empleados para que lo llenen?
  - e) Cree el cuestionario de encuesta, usando los lineamientos para escribir preguntas dados en este capítulo.
4. Basado en las respuestas para su encuesta de la intranet, usted considera que sería de ayuda entrevistar alguien en otra organización que haya tenido experiencia en el desarrollo, mantenimiento o ambos de una intranet corporativa.
- a) ¿Qué tipo de entrevista piensa usted que sería más apropiado en esta situación: no estructurada o estructurada? ¿Por qué?
  - b) Haga una cita con el administrador de la intranet en su empresa u otra organización o escuela para discutir sus experiencias en el desarrollo, mantenimiento o ambos de una intranet. Describa la organización y su intranet.
  - c) Prepare un guión de entrevista usando el formato de la figura 5-3 por poner un ejemplo, asegurándose que las preguntas no tienen los problemas discutidos en este capítulo.
  - d) Conduzca la entrevista, y registre las respuestas.
  - e) ¿Qué es lo que usted piensa que funcionó bien en la entrevista, y qué haría usted en forma diferente la próxima vez?
5. El lenguaje corporal es una parte sumamente importante de la comunicación, como se describe en este libro. Los analistas necesitan darse cuenta no sólo de lo que está siendo comunicado a través del lenguaje corporal del entrevistado, sino también del impacto que su lenguaje corporal puede tener sobre el proceso de la entrevista. Haga una cita con varios compañeros de trabajo o condiscípulos para hacer una entrevista en relación con las características que les gustaría ver en una intranet; si es posible, seleccione a los entrevistados que usted conoce bien y a esos que usted no conoce. Prepárese para las entrevistas siguiendo los mismos pasos de la pregunta anterior.
- a) Describa a los entrevistados que usted seleccionó y las preguntas que usted formuló.
  - b) Durante cada entrevista, observe las expresiones faciales del entrevistado. ¿Qué observó usted? ¿Eran consistentes siempre las expresiones faciales con las respuestas?
  - c) Durante cada entrevista, observe el contacto visual. ¿Cuánto tiempo duró? Observe y describa lo que sucedió cuando usted hizo contacto visual durante más de tres a cinco segundos con el entrevistado.
  - d) Pruebe cambiar su zona espacial durante la entrevista. ¿Mostró el entrevistado algún síntoma de sentirse incómodo? ¿En qué momento ocurrió esto?
  - e) ¿Notó usted alguna diferencia en el lenguaje corporal entre quienes usted conoce bien y quienes no conoce?
  - f) ¿Qué fue lo más exitoso y lo menos exitoso que hizo al suscitar información?
6. Comúnmente los analistas tienen acceso a datos confidenciales o sensibles durante la fase de identificación de requerimientos de un proyecto, en particular durante la exploración. Los analistas necesitan darse cuenta de situaciones donde pueden infringir la ética profesional, ya sea por actos de comisión u omisión, y las posibles consecuencias. Busque en la Web y/o en periódicos de negocios comerciales en la biblioteca de su escuela artículos sobre incidentes que involucren brechas de ética profesional.
- a) ¿Qué artículos encontró usted?
  - b) ¿Cuál fue la naturaleza de cada uno de estos incidentes?
  - c) ¿Cuáles fueron las consecuencias?
  - d) ¿Cuál fue la responsabilidad personal del analista en cada incidente?
  - e) ¿Qué pudo haberse hecho al nivel de la organización, individual o ambos para evitar el incidente o reducir su gravedad?

## Casos breves



1. En el capítulo 4, usted desarrolló estudios de factibilidad para un proyecto. Las evaluaciones económicas de factibilidad son afectadas significativamente por intangibles, cuyo valor se obtiene en parte por entrevistas y cuestionarios. Desarrolle preguntas de entrevista para determinar el valor para los empleados del trabajo a distancia.
  - a) Comience con preguntas no estructuradas planteadas a un grupo de empleados para determinar qué es lo que interesa a los empleados y cómo consideran el trabajo a distancia.
  - b) Una vez que usted sabe qué aspectos rodean a la percepción del empleado acerca del trabajo a distancia y por qué les podría agradar/desagradar, cree preguntas abiertas, pero que sean estructuradas, sobre esos asuntos, y entreviste a un segundo conjunto de trabajadores. ¿Por qué usamos dos grupos diferentes de empleados para este proceso?
2. Desarrolle un cuestionario para distribuirse masivamente entre los empleados, basado en sus conclusiones de las entrevistas previas. ¿Por qué estamos nosotros completando el análisis con una encuesta anónima?
3. Usted se encarga de desarrollar un sistema de inscripción de clase en línea nuevo para su escuela. Desarrolle un conjunto de preguntas de entrevista para determinar los asuntos y las necesidades de los estudiantes, el equipo de inscripción, y el profesorado para un sistema de inscripción en línea.
4. Discuta el impacto que las preguntas sesgadas o principales pueden tener sobre un análisis. Cree una pregunta de entrevista sin sesgo y una pregunta capciosa o sesgada. Plantee cada una de esas preguntas a cinco personas. ¿Qué clase de respuestas obtuvo? ¿Eran las que usted esperaba?

## Ejercicios de equipo e individuales



1. Cree un conjunto de preguntas de entrevistas con sesgo, capciosas o cargadas. Plantéelas a otro estudiante en la clase. El otro estudiante, en lugar de contestar las preguntas, le deberá decir cómo está sesgado usted y qué respuesta anda usted buscando.
2. Ejercicio de clase: Cree un conjunto de preguntas de entrevista que tengan el menor sesgo posible sobre un tema particular. Plantee las preguntas al grupo. Sin embargo, traiga puesta una playera, alfileres, etcétera, para conducir al grupo a que responda de una manera específica. Diviértase, y ex-

perimente con medios visuales, apoyos y cosas por el estilo.

3. Se ha encontrado en estudios de investigación pasados que los empleados a quienes se les permite el trabajo a distancia, en realidad trabajan aproximadamente tres horas extras sin pago a la semana. Pero el trabajo a distancia es a menudo utilizado como una herramienta negociadora por un empleador: para que puedan optar por el trabajo a distancia, los empleados deben aceptar un sueldo inferior, típicamente 10 por ciento. ¿Qué piensa usted acerca de esto?

## Lecturas recomendadas



- Andrews, D. C. y N. S. Leventhal. *Fusion Integrating IE, CASE and JAD: A Handbook for Reengineering the Systems Organization*. Englewood Cliffs, NJ: Prentice Hall, 1993.
- Berdie, Douglas R. y John F. Anderson. *Questionnaires: Design and Use*. Metuchen, NJ: Scarecrow Press, 1974. Una guía práctica para la redacción de cuestionarios. Especialmente útil debido a su pequeño tamaño y sus ejemplos ilustrativos.
- Davis, William S. *Systems Analysis and Design*. Reading, MA: Addison-Wesley, 1983. Suministra sugerencias útiles para la preparación y conducción de entrevistas.
- Dejoie, Roy; George Fowler y David Paradice. *Ethical Issues in Information Systems*. Boston, MA: Boyd and Fraser, 1991. Se centra en el impacto de la tecnología de las computadoras sobre la

toma de decisiones ética en las organizaciones de negocios actuales.

- Fitzgerald, Jerry; Ardra F. Fitzgerald y Warren D. Stallings, Jr. *Fundamentals of Systems Analysis*, 2a. ed. New York: John Wiley & Sons, 1981. Un útil libro de estudio para el analista de sistemas. El capítulo 6, "Understanding the Existing System", hace una muy buena presentación de las técnicas de exploración en la fase de estudio.

Gane, C. *Rapid Systems Development*. Nueva York: Rapid Systems Development, Inc., 1987. Este libro suministra una buena discusión sobre cómo conducir una reunión de grupo/entrevista.

Gause, Donald C., y Gerald M. Weinberg. *Exploring Requirements: Quality before Design*. Nueva York: Dorset House Publishing,

1989. Un excelente libro que describe las técnicas de la exploración de requerimientos.
- Gildersleeve, Thomas R. *Successful Data Processing System Analysis*. Englewood Cliffs, NJ: Prentice Hall, 1978. El capítulo 4, "Interviewing in Systems Work", da una visión completa de la entrevista específicamente para el analista de sistemas. En este capítulo se presenta un guión de una entrevista de muestra y se le analiza.
- London, Keith R. *The People Side of Systems*. Nueva York: McGraw-Hill, 1976. El capítulo 5, "Investigation versus Inquisition", ofrece una visión muy buena orientada hacia la gente de la exploración, con un énfasis considerable en las entrevistas.
- Lord, Kenniston W., Jr. y James B. Steiner. *CDP Review Manual: A Data Processing Handbook*, 2a. ed. New York: Van Nostrand Reinhold, 1978. El capítulo 8, "Systems Analysis and Design", presenta una comparación comprensiva de los méritos y los deméritos de cada técnica de exploración. Este material está destinado a la preparación de los procesadores de datos para los exámenes del Certificado de Procesamiento de Datos (Certificate in Data Processing, CDP), uno de los cuales cubre el análisis y diseño de sistemas.
- Miller, Irwin y John F. Freund. *Probability and Statistics for Engineers*. Englewood Cliffs, NJ: Prentice Hall, 1965. Un libro introductorio a nivel universitario sobre probabilidad y estadística.
- Mitchell, Ian; Norman Parrington; Peter Dunne y John Moses. "Practical Prototyping, Part One", *Object Currents*, mayo de 1996. El primero de una serie de artículos en tres partes que explora la realización de prototipos y cómo obtener un beneficio de ello. La realización de prototipos es una parte integral de JRP.
- Robertson, Suzanne y James Robertson. *Mastering the Requirements Process*. Reading, MA: ACM Press/Addison-Wesley, 1999. Este libro contiene una cobertura a gran profundidad de los procedimientos paso a paso de la identificación de requerimientos.
- Salvendy, G., ed. *Handbook of Industrial Engineering*. Nueva York: John Wiley & Sons, 1974. Un manual comprensivo para ingenieros industriales; en cierto modo los analistas de sistemas son un tipo de ingeniero industrial. Una cobertura excelente del muestreo y de la medición del trabajo.
- Stewart, Charles J. y William B. Cash, Jr. *Interviewing: Principles and Practices*, 2a. ed. Dubuque, IA: Brown, 1978. Un popular libro a nivel universitario que provee una amplia exposición de las técnicas de la entrevista, muchas de las cuales son aplicables al análisis y diseño de sistemas.
- Walton, Donald. *Are You Communicating? You Can't Manage without It*. Nueva York: McGraw-Hill, 1989. Este libro es una guía fácil de usar sobre el proceso de las comunicaciones y es obligatorio para cualquiera que trabaje con las personas y las influya.
- Weinberg, Gerald M. *Rethinking Systems Analysis and Design*. Boston: Little, Brown and Company, 1982. Un libro creado para estimular una nueva manera de pensar.
- Wood, Jane y Denise Silver. *Joint Application Design*. Nueva York: John Wiley & Sons, 1989. Este libro provee una visión general completa de la técnica de diseño conjunto de aplicaciones de IBM.



# Tema 6

Especificaciones de  
requerimientos con casos  
de uso



# 6 Modelado de requerimientos del sistema con los casos de uso

En este capítulo usted aprenderá acerca de las herramientas y las técnicas necesarias para la elaboración de modelos de requerimientos de sistemas usando casos de uso. La captura y la documentación de los requerimientos del sistema han demostrado ser factores críticos para el resultado de un proyecto exitoso de desarrollo de requerimientos de sistemas. La documentación de los requerimientos desde la perspectiva de los usuarios de manera que puedan entender, promueve la participación del usuario, lo que aumenta en gran medida la probabilidad de éxito del proyecto. Sabrá que entiende la modelación de los casos de uso de los requerimientos cuando usted pueda:

- Describir los beneficios de la modelación de los casos de uso.
- Definir a los actores y a los casos de uso y ser capaz de identificarlos de los diagramas de contexto y de otras fuentes.
- Describir los cuatro tipos de actores.
- Describir las relaciones que pueden aparecer en un diagrama de caso de uso.
- Describir los pasos para preparar un modelo de caso de uso.
- Describir cómo construir un diagrama de modelo de caso de uso.
- Describir las diversas secciones de una narración de caso de uso y ser capaz de preparar una.
- Definir el propósito de la jerarquización de los casos de uso y de la matriz de prioridades así como el diagrama de dependencia de los casos de uso.

## Introducción

Después de la reunión de planeación conjunta de requerimientos (*joint requirements planning*, JRP), que se realizó como una tarea de la fase de análisis de requerimientos, el equipo del proyecto del sistema de Servicios para miembros SoundStage, ha construido una lista de casos de uso que especifica la totalidad de la funcionalidad requerida del sistema. Primero, cada caso de uso era simplemente una frase verbal (tal como “Colocar una nueva orden”) que describía algo que uno o más usuarios querían hacer con el sistema. Luego, cada caso de uso se documentaba con una narración que describía en detalle la interacción deseada entre el usuario y el sistema. Entonces Bob Martínez y otros analistas de sistemas sostuvieron una serie de entrevistas con los usuarios para verificar estas narraciones de casos de uso. Finalmente, ellos están analizando cuáles son los casos que tienen la más alta prioridad para el sistema. La jefa de Bob, Sandra, dice que ella va a identificar para ellos cuál funcionalidad tiene que incluirse en el primer ciclo de construcción del sistema. El plan es llevar a estos casos de uso de la más alta prioridad al diseño lógico y fases posteriores e implantar una versión 1.0 (que funcione) del sistema a tiempo y dentro del presupuesto.

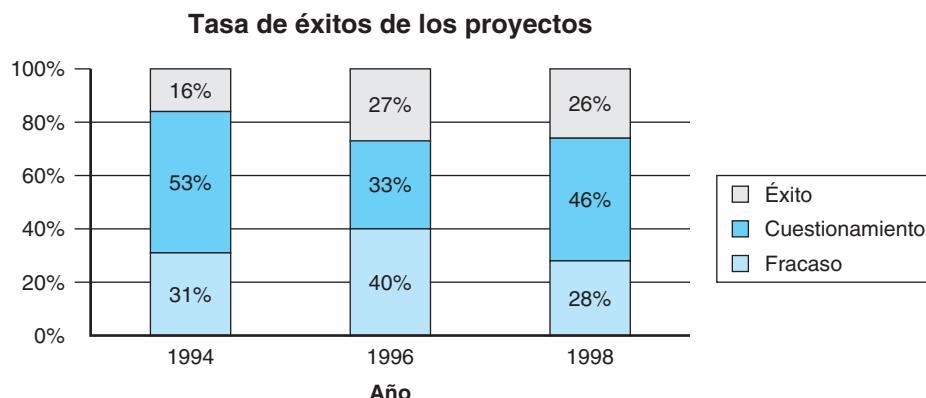
## Una introducción a la modelación de casos de uso

Uno de los retos fundamentales de importancia vital para cualquier equipo de desarrollo de sistemas de información, y especialmente para el analista de sistemas, es la capacidad para obtener, de los involucrados, los requerimientos correctos y necesarios del sistema, y especificarlos de manera inteligible para los involucrados de modo que estos requerimientos se verifiquen y se validen. De hecho, éste ha sido el caso durante muchos años, como lo escribió el distinguido autor Fred Brooks en su famoso artículo de 1987:

La parte individual más difícil de la construcción de un sistema de software es decidir con precisión qué construir. Ninguna otra parte del trabajo conceptual es tan difícil como establecer los requerimientos técnicos detallados, incluyendo todas las interfaces con las personas con las máquinas y con otros sistemas de software. Ningún otro trabajo invalida tanto al sistema resultante si se hace mal. Ninguna otra parte es más difícil de rectificar posteriormente.

El personal de tecnología de información siempre ha tenido problemas al tratar de especificar los requerimientos, especialmente los funcionales, de los usuarios. En el pasado hemos usado herramientas tales como los modelos de datos, los modelos de procesos, los prototipos, y las especificaciones de requerimientos que entendíamos y nos sentíamos cómodos con ellos, aunque eran difíciles de entender para cualquier usuario que no tuviera un entrenamiento en las prácticas de desarrollo de software. Debido a esto, muchos proyectos de desarrollo estaban, y todavía están, plagados con problemas de: menoscabo del alcance, sobregiro de costos y retraso del programa. Con frecuencia se desarrollan y se instalan sistemas que realmente no satisfacen las necesidades de los usuarios. Algunos se archivan y no se usan nunca, y un gran porcentaje se cancela aun antes de terminar el esfuerzo de desarrollo. Una compañía de investigación muy conocida, el Standish Group, estudió 23 000 aplicaciones de TI en 1994, 1996 y 1998.<sup>1</sup> Como se muestra en la figura 6.1, el estudio de 1998 encontró que solamente un poco más de un cuarto de los proyectos en 1998 tuvieron éxito (dentro del presupuesto, a tiempo, y cubría todas las necesidades). Más de un cuarto de ellos fracasó (se cancelaron antes de terminarse). Un poco más de la mitad fueron lo que Standish consideraba como cuestionables: el proyecto estaba terminado y operaba, pero se terminó ya sea con un sobregiro en el presupuesto, rebasando el tiempo estimado, o sin todas las características especificadas por los usuarios. La buena noticia que se refleja en estos estudios y en otros es que las prácticas y los métodos que estamos usando para desarrollar los sistemas de información están mejorando. La industria de desarrollo de software ha aprendido que con objeto de planear, analizar, diseñar,

<sup>1</sup> The Standish Group International, Inc., “CHAOS: A Recipe for Success” (versión electrónica), 1999. Obtenido el 5 de diciembre de 2002, de [www.pm2go.com/sample\\_research/chaos1998.pdf](http://www.pm2go.com/sample_research/chaos1998.pdf). El Standish Group es reconocido por su independiente investigación y análisis de la industria de tecnología de la información.

**FIGURA 6.1**

Tasas de éxito de los proyectos tal como lo reporta el grupo Standish

Fuente: The Standish Group International, Inc., "Chaos: A Recipe for Success" (versión electrónica), 1999, [www.pm2go.com/sample\\_research/chaos1998.pdf](http://www.pm2go.com/sample_research/chaos1998.pdf)

construir e implantar un sistema de información con éxito, el analista de sistemas debe entender primero las necesidades de los involucrados y las razones por las cuales debe desarrollarse el sistema; un concepto llamado **desarrollo centrado en el usuario**. Al centrarse en los usuarios del sistema, el analista puede concentrarse en cómo se va a usar el sistema y no en cómo va a construirse. La **modelación de casos de uso** es un enfoque que facilita el desarrollo centrado en el usuario.

La modelación de casos de uso tiene sus raíces en la modelación orientada a objetos, y usted aprenderá más acerca de cómo aplicar la modelación de casos de uso en el análisis orientado a objetos, pero también ha ganado popularidad en los ambientes de desarrollo que no son orientados a objetos. Usted aprenderá en los capítulos restantes de este libro cómo la modelación de casos de uso complementa al análisis tradicional de sistemas y a las herramientas de diseño tales como la modelación de datos y la modelación de procesos y al mismo tiempo, proporciona una base para las decisiones de arquitectura y las decisiones de diseño de interfaces.

La modelación de casos de uso fue concebida originalmente por el Dr. Ivan Jacobson en 1986 y adquirió popularidad tras la publicación de su libro, *Object-Oriented Software Engineering*, en 1992. El Dr. Jacobson usó la modelación de casos de uso como marco para su metodología de objetos; él la usó con mucho éxito para desarrollar los sistemas de información orientados a objetos. La modelación de casos de uso ha resultado ser una valiosa ayuda para enfrentar los retos de determinar cuál sistema se requiere hacer desde la perspectiva del usuario y del involucrado. En la actualidad se le reconoce ampliamente como la mejor práctica para la definición, la documentación y la comprensión de los requerimientos funcionales de un sistema de información.

El uso de la modelación de casos de uso facilita y alienta la participación del usuario, que es uno de los principales factores críticos de éxito para asegurar el éxito del proyecto. Además, la modelación de casos de uso proporciona los siguientes beneficios:

- Proporciona una herramienta para capturar los requerimientos funcionales.
- Ayuda a descomponer el alcance del sistema en piezas más manejables.
- Proporciona un medio de comunicación con los usuarios y con otros involucrados en relación con la funcionalidad del sistema. Los casos de uso presentan un lenguaje común que se entiende fácilmente por los diferentes involucrados.
- Proporciona un medio para identificar, asignar, rastrear, controlar y administrar las actividades de desarrollo de sistemas, especialmente un desarrollo por incrementos e iterativo.
- Proporciona una ayuda para estimar el alcance de proyecto, el esfuerzo a realizar y la programación.
- Proporciona una línea de base para pruebas en términos de la definición de los planes y casos de prueba.
- Proporciona una línea de base para los sistemas y manuales que ayudan al usuario así como para la documentación de desarrollo del sistema.
- Proporciona una herramienta para el seguimiento de los requerimientos.
- Proporciona un punto inicial para la identificación de los objetos o entidades de datos.
- Proporciona especificaciones funcionales para el diseño de las interfaces entre el usuario y el sistema.
- Proporciona un medio para definir los requisitos de acceso a la base de datos en términos de crear, cambiar, borrar y leer.
- Proporciona un marco para impulsar el proyecto de desarrollo de sistemas.

**desarrollo centrado en el usuario** Proceso de desarrollo de sistemas basado en la comprensión de las necesidades de los involucrados y las razones por las que deben desarrollarse los sistemas.

**modelación de casos de uso** Proceso de modelación de las funciones de un sistema en términos de eventos de negocios, quiénes iniciaron los eventos y cómo responde el sistema a estos eventos.

## Conceptos de sistemas en la modelación de casos de uso

## diagrama de casos de

**uso** Diagrama que ilustra las interacciones entre el sistema y los sistemas y usuarios externos. En otras palabras, describe gráficamente quién va a usar el sistema y de qué manera el usuario espera interactuar con el sistema.

**descomposición funcional** Acto de desarmar un sis-

**Analizar** Acto de desarmar un sistema en subcomponentes.

## **narración del caso de**

**uso** Texto que describe el evento del negocio y la forma en que el usuario interactuará con el sistema para lograr la tarea.

**casos de uso** Una secuencia de pasos relacionados (un escenario), tanto automatizado como manual, con el propósito de completar una sola tarea del negocio.



## **FIGURA 6.2**

## Un ejemplo de un diagrama de modelo de casos de uso

Existen dos elementos primordiales cuando se realiza la modelación de casos de uso. El primero es el **diagrama de casos de uso**, que ilustra gráficamente al sistema como una colección de casos, actores (usuarios) y sus relaciones. Este diagrama comunica a un alto nivel el alcance de los eventos de negocios que el sistema debe procesar. En la figura 6.2 se muestra un ejemplo de un diagrama de casos de uso. Muestra cada una de las funciones del sistema, o eventos del negocio (dibujados en las elipses), y a los actores, o usuarios del sistema, quienes interactúan con esas funciones. Como usted ve en la figura 6.2, los actores pueden situarse a ambos lados del conjunto de las figuras de casos de uso e interactuar con uno o más de los casos. El diagrama de casos de uso es muy sencillo. Pero con éste, comienza un importante proceso llamado **descomposición funcional**, el acto de descomponer un sistema en sus subcomponentes. Es imposible entender inmediatamente el sistema completo, pero es posible entender y especificar cada parte del mismo.

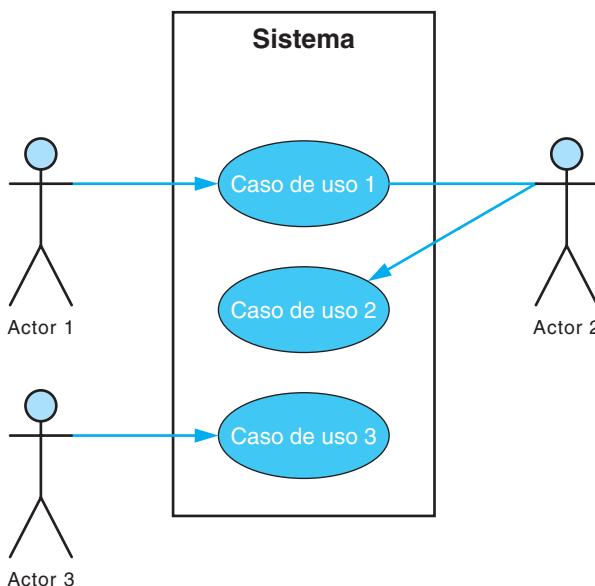
El segundo elemento, llamado la **narración del caso de uso**, describe los detalles de cada evento de negocios y especifica cómo interactúan los usuarios con el sistema durante ese evento. La narración del caso de uso se estudiará con detalle más adelante en el capítulo.

## > Los casos de uso

La modelación de los casos de uso identifica y describe las funciones del sistema mediante el uso de una herramienta llamada **casos de uso**. Dichos casos describen las funciones del sistema desde la perspectiva de los usuarios externos de una forma y con una terminología que ellos entienden. El alcanzar exacta y minuciosamente esto requiere un alto nivel de participación por parte del usuario, así como de un experto en la materia que sea versado en el proceso o evento de negocios.

Los casos de uso se representan gráficamente con una elipse horizontal con el nombre del caso que aparece encima, debajo o dentro de la elipse. Un caso de uso representa un objetivo individual del sistema y describe una secuencia de actividades y de interacciones del usuario para tratar de alcanzar el objetivo. La creación de los casos de uso ha probado ser una técnica excelente para entender y documentar mejor los requerimientos del sistema. Un caso de uso por sí solo no se considera como un requerimiento funcional, pero la historia (el escenario) que relata el caso de uso consiste en uno o más requerimientos.

Inicialmente, los casos de uso se definen durante la etapa de los requerimientos del ciclo de vida y se refinarán adicionalmente a lo largo del ciclo de vida. Durante la iden-



tificación de los requisitos, los casos de uso se emplean para capturar la esencia de los problemas de negocios y para modelar (a un alto nivel) la funcionalidad del sistema propuesto. Adicionalmente, son el punto inicial para la identificación de las entidades de datos (cubierto en el capítulo 7) o de los objetos del sistema (cubierto en el capítulo 9). Durante el análisis de requisitos, los casos de uso se refinan para modelar el uso del sistema con más detalle. En otras palabras, se actualizan para especificar lo que los usuarios están tratando de alcanzar y el porqué. Estos casos ayudan en la definición de los prototipos o interfaces del usuario. Durante el diseño los casos de uso se refinan para modelar cómo los usuarios usarán realmente el sistema con respecto a cualquier interfaz y a las restricciones del sistema. Estos tipos de casos de uso ayudan a identificar el comportamiento del objeto o del sistema y a diseñar las especificaciones de código y de interfaz, así como para servir para planear las pruebas del sistema. En la construcción, los casos de uso ayudan a los desarrolladores en la programación y en las pruebas. Estos casos también sirven como la línea base para preparar cualquier documentación del sistema y para el usuario, además de servir como herramientas para el entrenamiento de los usuarios. Y como los casos de uso contienen una enorme cantidad de detalle de la funcionalidad del sistema, éstos serán un recurso constante para validar al sistema.

## > Actores

Los casos de uso se inician o son generados por los usuarios externos llamados **actores**. Un actor inicia la actividad del sistema, un caso de uso, con el propósito de terminar alguna tarea de negocios que produzca algo con valor apreciable. Utilicemos el ejemplo de un estudiante universitario que se inscribe en los cursos del semestre de otoño. El actor sería el *estudiante*, y el evento de negocios, o caso de uso, sería *Inscribirse en Curso*. Un actor representa un papel desempeñado por un usuario que interactúa con el sistema y no significa que retrate a una persona o un puesto de trabajo. De hecho, un actor no tiene que ser humano. Puede ser una organización, otro sistema de información, un dispositivo externo tal como un sensor de calor, o aun el concepto de tiempo (que se estudiará un poco después). Un actor se representa gráficamente como una figura de línea rotulada con el nombre del papel que juega el actor.

Es importante observar que hay principalmente cuatro tipos de actores:

**actor** Cualquier cosa que necesite interactuar con el sistema para intercambiar información.



Símbolo de un actor

- **Actor primario de negocios:** el interesado que se beneficia principalmente de la ejecución de un caso de uso al recibir algo de valor medible u observable. El actor primario de negocios puede o no iniciar el evento de negocios. Por ejemplo, en el evento de negocios de un empleado que recibe un cheque como pago (algo con valor medible) del sistema de nómina cada viernes, el empleado no inicia el evento pero es el receptor primario del algo de valor.
- **Actor primario del sistema:** el involucrado que tiene una interfaz directa con el sistema para iniciar u ocasionalmente el evento de negocios o de sistema. Los actores primarios del sistema pueden interactuar con los actores primarios de negocios con el propósito de usar el sistema real. Ellos facilitan el evento a través del uso directo del sistema para beneficio del actor primario de negocios. Los ejemplos incluyen un dependiente de una tienda de abarrotes que selecciona los artículos para el cliente que compra abarrotes, una operadora de teléfonos que proporciona información del directorio a un cliente, y un cajero de banco que procesa una transacción bancaria. El actor principal de negocios y el actor principal de sistema pueden ser la misma persona para eventos en los cuales el actor de negocios tiene una interfaz directa con el sistema; por ejemplo, una persona que reserva la renta de un automóvil a través de un sitio Web.
- **Actor externo servidor:** el involucrado que responde a una solicitud desde el caso de uso (por ejemplo, un buró de crédito que autoriza el pago mediante tarjeta de crédito).
- **Actor externo receptor:** el involucrado que no es el actor primario pero que recibe algo de valor medible u observable (salida) proveniente del caso de uso (por ejemplo, un almacén que recibe una orden de embalaje para preparar un flete después de que un cliente ha colocado una orden).

En muchos sistemas de información hay eventos de negocios ocasionados por el calendario o la hora del reloj. Considere los siguientes ejemplos:

- El sistema de facturación de una compañía de tarjetas de crédito genera automáticamente sus estados de cuenta en el quinto día de cada mes (fecha de facturación).
- Un banco concilia sus transacciones con cheques todos los días a las 5 p.m.
- Cada noche se genera automáticamente un reporte que lista cuáles cursos están cerrados a las inscripciones (ya no hay más lugares) y cuáles cursos todavía están abiertos.

**evento temporal** Evento del sistema que es activado por el tiempo.

Estos eventos son ejemplos de **eventos temporales**. ¿Quién sería el actor? Todos los eventos listados antes se ejecutaron (o se ocasionaron) automáticamente: cuando se cumplió cierta fecha o cierta hora. Por eso decimos que el actor de un evento temporal es el tiempo.

## > Relaciones

Una relación se ilustra como una línea entre dos símbolos en el diagrama de casos de uso. El significado de las relaciones puede diferir dependiendo de cómo se dibujen las líneas y qué tipos de símbolos conectan. En las siguientes secciones definiremos las diferentes relaciones que se encuentran en un diagrama de casos de uso.

**asociación** Relación entre un actor y un caso de uso en la que interactúan entre sí.

**Asociaciones** Existe una relación entre un actor y un caso de uso siempre que el caso describa una interacción entre éstos. A esta relación se le denomina **asociación**. Como se indica en la figura 6.3, una asociación se modela como una línea continua que conecta al actor y al caso de uso. Una asociación que contiene una cabeza de flecha en el extremo que toca al caso de uso (1) indica que el caso fue iniciado por el actor en el otro extremo de la línea. Las asociaciones sin cabezas de flecha (2) indican una interacción entre el caso de uso y un actor externo servidor o receptor. Si un actor se asocia con un caso de uso, decimos que el actor se *comunica* con el caso. Las asociaciones pueden ser bidireccionales o unidireccionales.

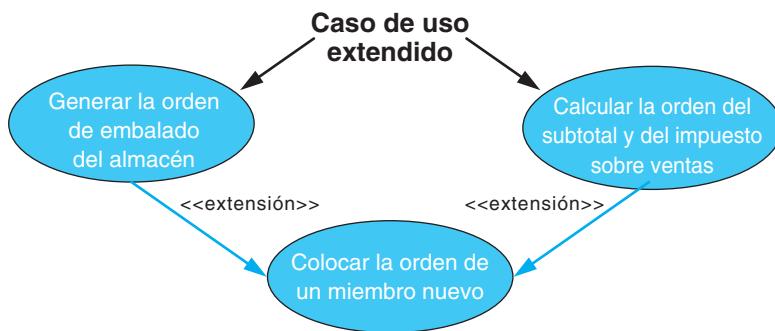
**caso de uso de extensión**  
Un caso de uso que consiste en los pasos extraídos de otro más complejo para simplificar el caso original y, así, ampliar su funcionalidad. El caso de uso de extensión de hecho extiende la funcionalidad del caso de uso original.

**Extensión** Un caso de uso puede contener una funcionalidad compleja que consiste de varios pasos que hacen difícil entender a la lógica del caso. Con objeto de simplificar el caso de uso y hacer que se entienda más fácilmente, podemos extraer los pasos más complejos para formar su propio caso. El caso resultante se llama un **caso de uso de extensión** ya que extiende la funcionalidad del caso de uso original. La relación entre el caso de uso de extensión y el que se está extendiendo se llama una relación de *extensión*. Un caso de uso puede tener muchas relaciones de extensión, pero un caso de uso de extensión puede ser invocado solamente por el caso que se esté extendiendo. Como se ilustra en la figura 6.4, la relación de extensión se representa como una línea con cabeza de flecha (ya sea continua o segmentada) que comienza en el caso de uso de extensión y que apunta al caso de uso que se está extendiendo. Cada línea de relación de extensión se rotula como “<<extensión>>”. Generalmente los casos de uso de extensión no se identifican en la fase de requerimientos, sino en la de análisis.

**FIGURA 6.3**

Ejemplo de una relación de asociación



**FIGURA 6.4**

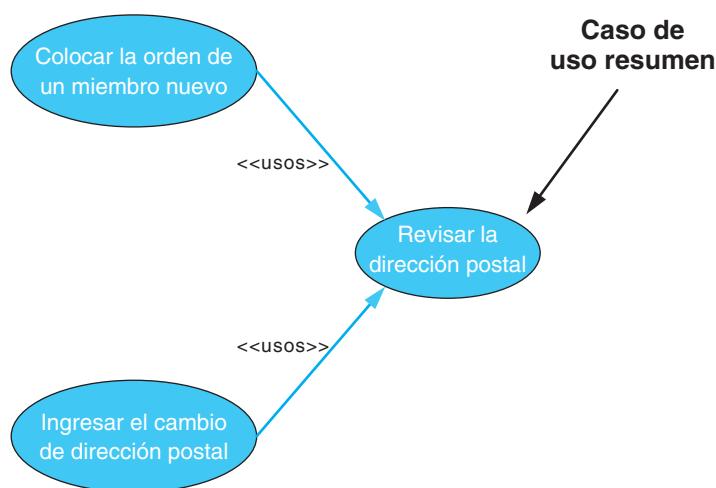
Ejemplo de una relación de extensión

**Usos (o inclusión)** Muy comúnmente, usted puede descubrir dos o más casos de uso que ejecuten pasos de funcionalidad idéntica. Lo mejor es extraer estos pasos comunes para formar un caso de uso separado que sea propio llamado un **caso de uso resumen**. Un caso de uso resumen representa una forma de “reuso” y es una herramienta excelente para reducir la redundancia entre los casos de uso. Un caso de uso resumen está disponible como referencia (o uso) para cualquier otro caso de uso que requiera su funcionalidad. La relación entre el caso de uso resumen y el caso de uso que lo usa se llama una relación de *uso* (algunas herramientas de la modelación de casos de uso lo denominan una relación de *inclusión*). La relación de uso que se presenta en la figura 6.5 se ilustra con una línea con cabeza de flecha (ya sea continua o segmentada) que comienza en el caso de uso oficial y que apunta al caso de uso que esté usando. Cada línea de relación de uso se rotula “<<uso>>”. Generalmente los casos de uso resúmenes no se identifican en la fase de requisitos, sino en la de análisis.

**caso de uso resumen** Un caso de uso que reduce la redundancia entre dos o más casos de uso al combinar los pasos comunes existentes en estos casos. Otro caso de uso *utiliza* o *incluye* el caso de uso resumen.

**Dependencia** Como administrador de proyecto o desarrollador líder, es de mucha ayuda saber cuáles casos de uso tienen una dependencia sobre otros casos de uso con objeto de determinar la secuencia en que es necesario desarrollar los casos de uso. Si usamos el negocio bancario como ejemplo, el caso de uso Hacer un retiro no puede ejecutarse hasta que haya ocurrido el caso de uso Abrir una cuenta bancaria. Debido a estas dependencias, el equipo de desarrollo muy probablemente escogerá desarrollar el caso de uso Abra una cuenta bancaria primero, en segundo lugar el caso de uso Haga un depósito, y en tercer lugar el caso de uso Haga un retiro para los propósitos de Condiciones de uso y pruebas. Un diagrama de casos de uso que modele las dependencias de caso de usos del sistema mediante el uso de la relación de **dependencia** proporciona un modelo que es una herramienta excelente para propósitos de planeación y de programación. La relación de dependencia tal como se presenta en la figura 6.6 se ilustra con una línea con cabeza de flecha

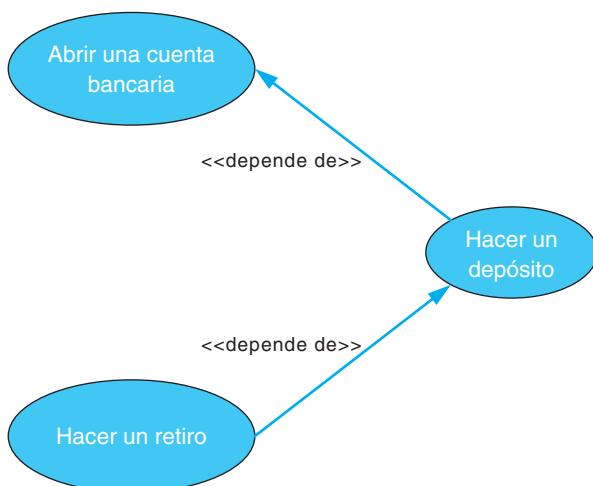
**dependencia** Relación entre casos de uso que indica que un caso de uso no puede realizarse hasta que se haya realizado el otro caso de uso.

**FIGURA 6.5**

Ejemplo de una relación de usos

**FIGURA 6.6**

Ejemplo de una relación de dependencia

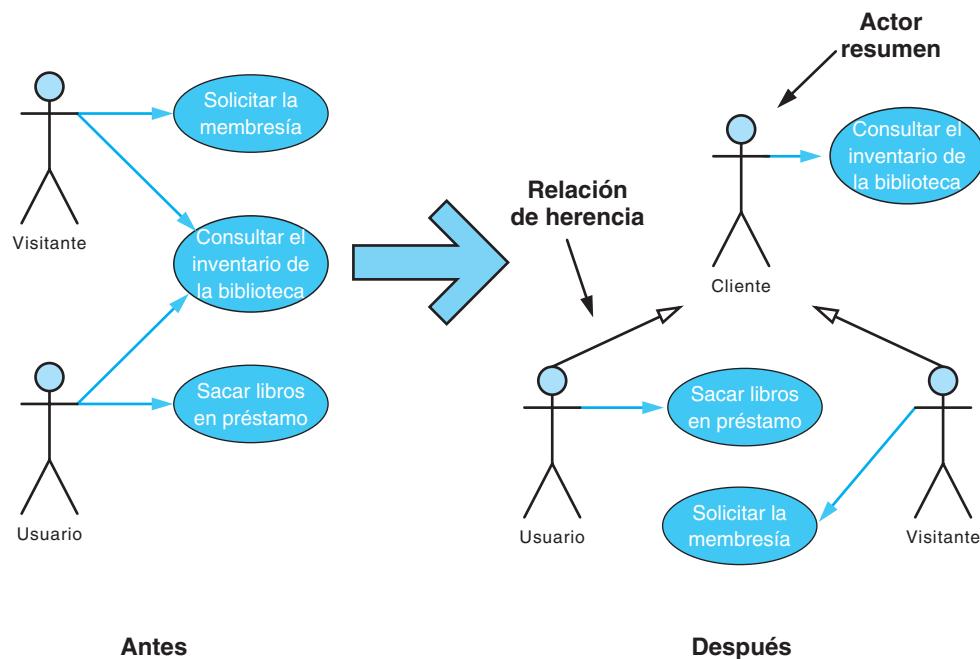


(ya sea continua o segmentada) que comienza en un caso de uso y que apunta al caso de uso del cual depende. La línea de relación de dependencia se rotula “<<depende de>>”.

**Herencia** Cuando dos o más actores comparten un comportamiento común (en otras palabras, pueden iniciar el mismo caso de uso) lo mejor es extrapolar este comportamiento común y asignarlo a un nuevo actor *resumen* con objeto de reducir la comunicación redundante con el sistema. Por ejemplo, un *usuario* de una biblioteca es un miembro que tiene una tarjeta que está autorizado a “Buscar en el inventario de la biblioteca” así como a “Sacar libros en préstamo” de la biblioteca. Como muchas bibliotecas son instituciones públicas, ellos dan la bienvenida a los *visitantes* para que usen sus servicios en el lugar tales como “Buscar en el inventario de la biblioteca” pero a los visitantes no se les permite la extensión de servicios (tal como “Sacar libros en préstamo”) que están reservados para los usuarios. Al crear un actor resumen llamado *cliente*, del cual van a heredar el *usuario* y el *visitante*, tenemos que modelar la relación solamente una vez iniciando el caso de uso Buscar en el inventario de la biblioteca. En el diagrama de caso de uso se ilustra la relación **herencia** con el tipo de flecha mostrado en la sección “Después” de la figura 6.7.

**FIGURA 6.7**

Ejemplo de una relación de herencia



## El proceso de la modelación de los casos de uso para los requerimientos

El objetivo de construir el modelo de casos de uso para los requerimientos es producir y analizar suficiente información de los requerimientos para preparar un modelo que comunique lo que se requiere desde la perspectiva del usuario pero que esté libre de los detalles específicos acerca de cómo va a construirse e implantarse el sistema. El seguimiento de este enfoque producirá después un diseño que es más robusto y que posiblemente sea menos impactado por el cambio. Pero para estimar y programar el proyecto con efectividad puede ser necesario que el modelo incluya “hipótesis de implantación de sistemas” preliminares para ayudar en esas actividades. Es crítico que el analista no se deslice a un estado de *parálisis del análisis* al preparar este modelo. La velocidad es la clave. No todos los hechos serán aprendidos durante esta fase del ciclo de vida, pero al utilizar un desarrollo iterativo y por incrementos, la metodología permite después la introducción de nuevos requerimientos en el proyecto sin impactar seriamente el desplegado de la solución final. Los pasos requeridos para producir este modelo son los siguientes:

1. Identificar a los actores de negocios.
2. Identificar los casos de uso para los requerimientos de negocios.
3. Construir un diagrama para el modelo de casos de uso.
4. Documentar las narraciones de casos de uso para los requerimientos de negocios.

### > Paso 1: Identificar a los actores de negocios

¿Por qué identificar primero a los actores? Al centrarse en los actores, usted puede concentrarse en cómo se usará el sistema y no en cómo se construirá. Además ayuda a refinar y definir aún más el alcance y las fronteras del sistema. Los actores también determinan que tan completos están los requerimientos del sistema.<sup>2</sup> Un beneficio de identificar a los actores primero es que al hacerlo se identifica a los candidatos que podemos entrevistar y observar posteriormente para terminar el esfuerzo de modelación de los casos de uso. Además, es posible usar a estas mismas personas para verificar y validar los casos de uso cuando terminan.

¿Dónde busca usted a los actores potenciales? Las siguientes referencias son fuentes excelentes:

- Un diagrama de contexto que identifique el alcance o las fronteras del sistema.
- La documentación del sistema y los manuales del usuario existentes.
- Minutas de las juntas y los talleres del proyecto.
- Documentos de los requerimientos, carta constitutiva del proyecto, o declaración del trabajo existente.

Al buscar actores haga las siguientes preguntas:

- ¿Quién o qué proporciona las entradas al sistema?
- ¿Quién o qué recibe las salidas del sistema?
- ¿Se requieren interfaces con otros sistemas?
- ¿Existen eventos que son originados automáticamente en un instante predeterminado?
- ¿Quién mantendrá la información en el sistema?

Los actores deberán nombrarse con un sustantivo o con una frase sustantiva.

Cuando usted identifique a un actor, cree el texto de una definición de ese actor de acuerdo con la perspectiva del usuario y usando sus términos. La figura 6.8 es una plantilla de un glosario de actores que puede usarse para documentar a los actores. Este ejemplo contiene un listado parcial de los actores del Sistema de Servicios para los Miembros de SoundStage.

<sup>2</sup> Frank Armour y Granville Miller, *Advance Use Case Modeling* (Boston: Addison-Wesley, 2001).

**FIGURA 6.8**

Lista parcial de los actores del Sistema de servicios para los miembros de SoundStage

**Glosario de actores**

Término	Sinónimo	Descripción
1. Miembro potencial		Una persona o corporación que ingresa una orden de suscripción con objeto de incorporarse al club.
2. Miembro del club	Miembro	Una persona o corporación que se adhiere al club mediante un convenio.
3. Ex miembro	Miembro inactivo	Un tipo de miembro que ha cumplido con la obligación del acuerdo, no ha colocado una orden en los últimos seis meses pero que todavía tiene derechos.
4. Mercadeo		Organización responsable de la creación de los programas de promoción y suscripción y de la generación de ventas para la compañía.
5. Servicios a los miembros		Organización responsable de suministrar un punto de contacto para los clientes de SoundStage Entertainment en términos de acuerdos y órdenes.
6. Centro de distribución	Almacén	Entidad que aloja y mantiene el inventario de productos de SoundStage Entertainment y que procesa los fletes y las devoluciones de los clientes.
7. Cuentas por cobrar		Organización responsable del procesamiento de los pagos y las facturas de los clientes así como de mantener la información de las cuentas de los clientes.
8. Tiempo		Concepto de actor responsable de ocasionar eventos temporales.

### > Paso 2: Identificar los casos de uso para los requerimientos de negocio

#### **caso de uso de requerimientos del negocio**

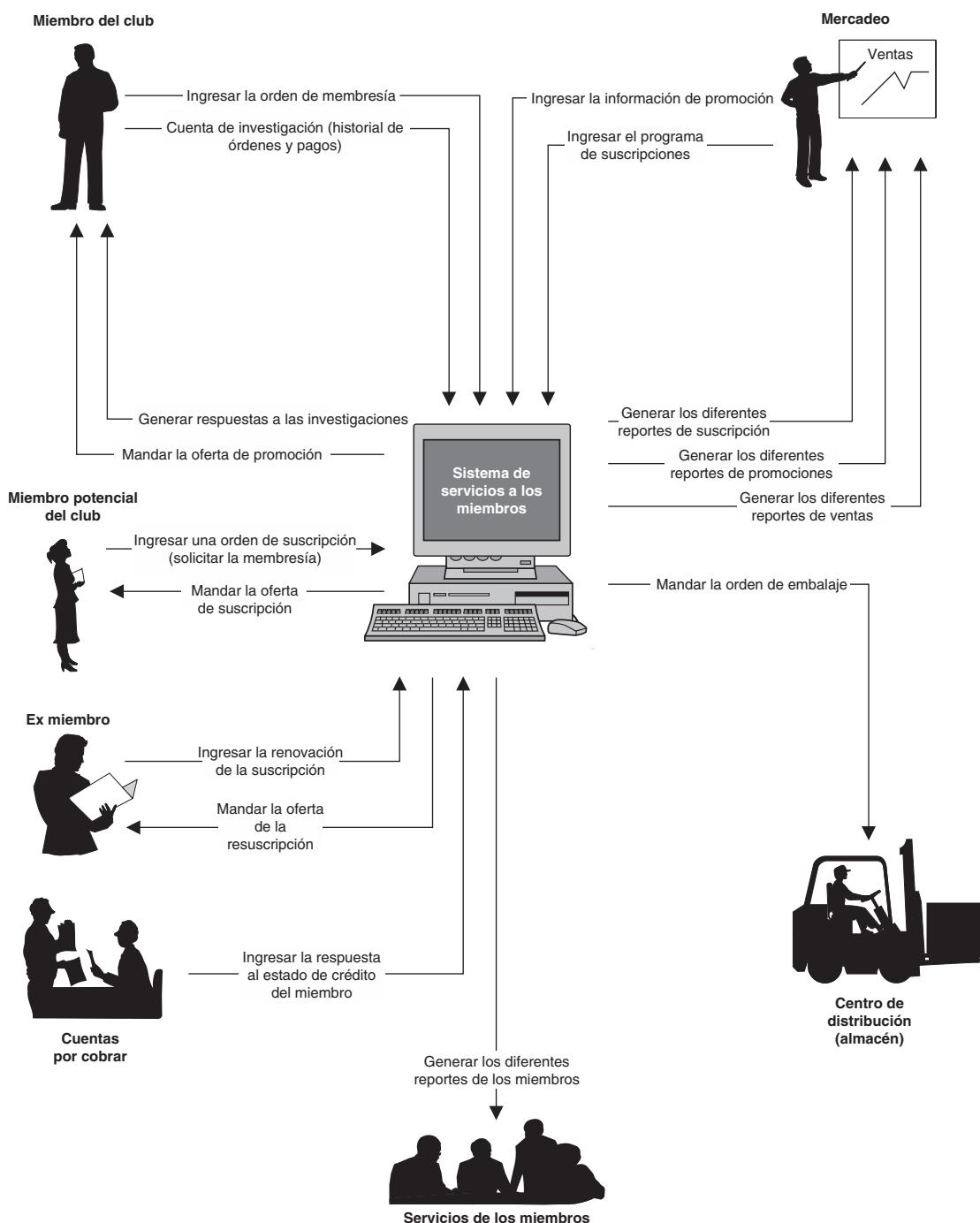
**Caso de uso** creado durante el análisis de requerimientos para capturar la interacción entre un usuario y el sistema con independencia de los detalles de tecnología e implantación, también llamado un caso de uso esencial.

Un sistema de información típico puede constar de docenas de casos de uso. Durante el análisis de requerimientos tratamos de identificar y documentar solamente los más críticos, complejos e importantes, frecuentemente denominados casos de uso *esenciales* debido a consideraciones de tiempo y de costo. Un **caso de uso de requerimientos del negocio** captura las interacciones con el usuario de manera que esté libre de los detalles de la tecnología y de la implantación. Como un caso de uso describe cómo interactúa un actor en el mundo real con el sistema, una técnica excelente para encontrar los casos de uso de los requerimientos de negocios es examinar a los actores y cómo van a usar al sistema. Al buscar los casos de uso haga las siguientes preguntas:

- ¿Cuáles son las principales tareas del actor?
- ¿Qué información necesita el actor del sistema?
- ¿Qué información proporciona el actor al sistema?
- ¿Necesita el sistema informar al actor de cambios o eventos que hayan ocurrido?
- ¿Necesita informar el actor al sistema de cambios o eventos que hayan ocurrido?

Nuevamente, un diagrama de contexto es una fuente excelente para encontrar casos de uso potenciales. Los diagramas de contexto se estudiaron en el capítulo 4. Vienen de la modelación tradicional de procesos (capítulo 8) pero son útiles aun para proyectos que adoptan un enfoque orientado a objetos. Examinemos el diagrama de contexto del Sistema de Servicios para Miembros de SoundStage en la figura 6.9. Podemos identificar los casos de uso potenciales al examinar el diagrama e identificar las entradas y salidas primarias del sistema y las partes externas que las ingresan y las reciben. Las entradas primarias que ocasionan los eventos de negocios (por ejemplo, *Ingresar la orden del miembro*) dentro de la organización se consideran casos de uso, y las partes externas que suministran estas entradas se consideran actores (por ejemplo, *Miembro del club*). Es importante observar que las entradas que son el resultado de peticiones al sistema no indican un

### Diagrama de contexto de los servicios para los miembros



**FIGURA 6.9** Diagrama de contexto del Sistema de servicios para los miembros de SoundStage

caso de uso separado; y tal como una compañía de tarjetas de crédito que responde a una solicitud de autorización o, como se presenta en la figura 6.9, el actor *Cuentas por cobrar* que responde con *Información del nivel crediticio del miembro*.

Los casos de uso se denominan con un enunciado que especifica la meta del actor, tal como *Ingresar una orden de suscripción*. Los casos de uso que son eventos temporales,

generalmente se identifican como el resultado de analizar las salidas clave del sistema. Por ejemplo, cualquier salida que se genere basada en el tiempo o en una fecha, tales como los reportes mensuales o anuales, se considera como un caso de uso, y el actor, como usted recuerda, es el tiempo. En la figura 6.9 supongamos que uno de los diferentes reportes que recibe Servicios para los miembros es un *Reporte de conformidad por omisión a 10-30-60 días* que se genera automáticamente de manera cotidiana. Como la generación del reporte es originada por el tiempo se requiere un caso de uso para procesar el evento, y lo denominaríamos *Reporte de conformidad por omisión a 10-30-60 días generado rutinariamente*. Es importante observar que muchas veces los reportes individuales no se listan en un diagrama de contexto porque son demasiado numerosos y podrían saturar el diagrama y hacerlo difícil de leer. Es la responsabilidad del analista de sistemas investigar con los involucrados apropiados el tipo de salidas que reciben y sus características, en términos de volumen, frecuencia y mecanismo de accionamiento, con objeto de identificar a los “casos de uso ocultos”.

La figura 6.10 es una plantilla de un glosario de casos de uso que puede usarse para documentarlos. Este ejemplo contiene una lista parcial de casos de uso y de los actores del Sistema de servicios para los miembros de SoundStage identificados del diagrama de contexto así como de otras fuentes.

### > Paso 3: Construir el diagrama del modelo de casos de uso

Una vez que los casos de uso y los actores han sido identificados, puede usarse un diagrama de modelo de casos de uso para ilustrar gráficamente el alcance y las fronteras

**FIGURA 6.10** Lista parcial de los casos de uso del Sistema de servicios para los miembros de SoundStage

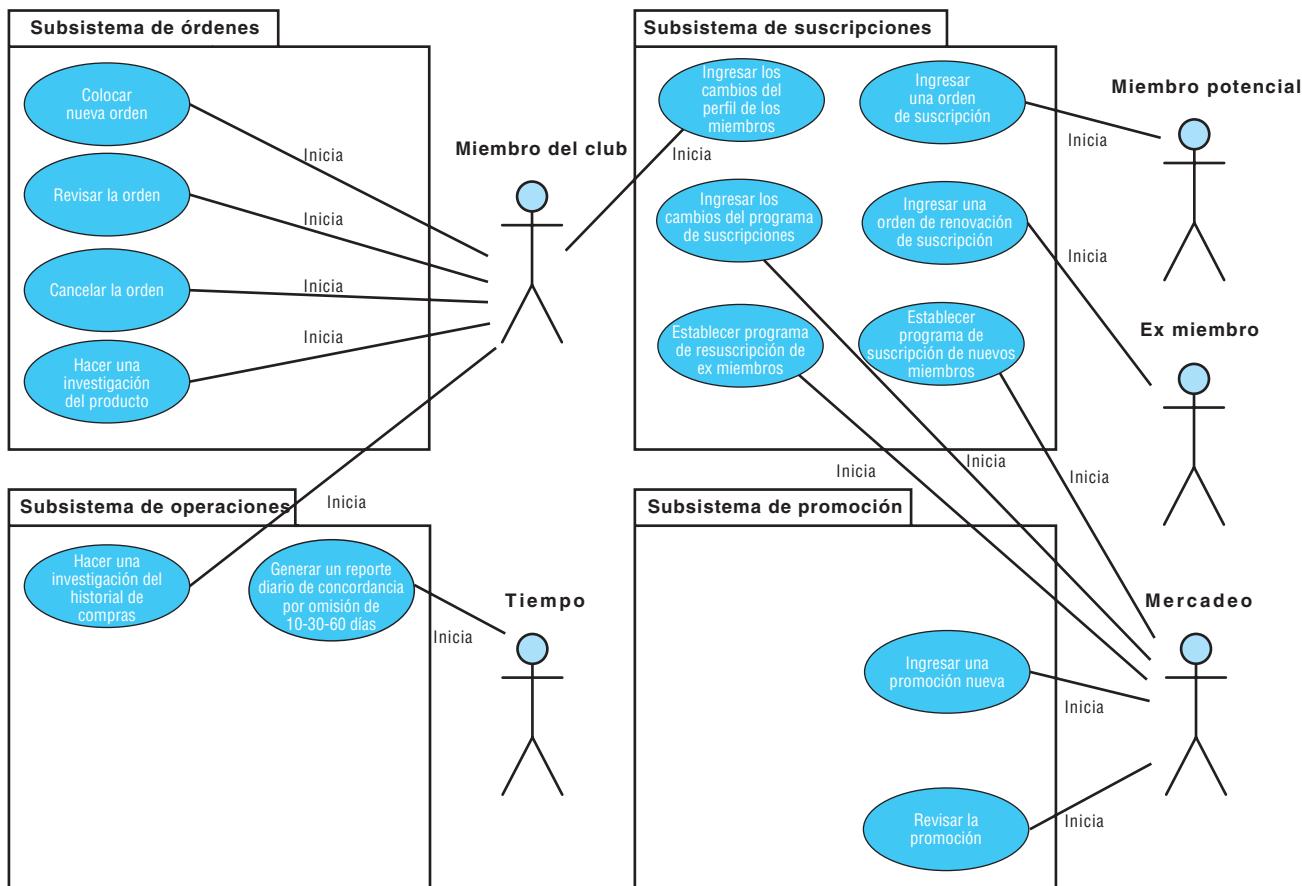
#### Glosario de casos de uso

Nombre del caso de uso	Descripción del caso de uso	Actores y papeles participantes
Ingresar una orden de suscripción	Este caso de uso describe el evento de un miembro potencial que solicita ingresar al club por suscripción. (“Tome un paquete de 12 CD por un chelín y concuerde en comprar cuatro más a precios regulares en menos de dos años.”)	<ul style="list-style-type: none"> <li>Miembro potencial (negocio primario)</li> <li>Centro de distribución (receptor externo)</li> </ul>
Ingresar una orden de renovación de suscripción	Este caso de uso describe el evento de un ex miembro que solicita reingresar al club por suscripción. (“Tome un paquete de 12 CD por un chelín y concuerde en comprar cuatro más a precios regulares en menos de dos años.”)	<ul style="list-style-type: none"> <li>Ex miembro (negocio primario)</li> <li>Centro de distribución (receptor externo)</li> </ul>
Ingresar los cambios del perfil del miembro	Este caso de uso describe el evento de un miembro del club que ingresa cambios de su perfil en cosas tales como la dirección postal, la dirección de e-mail, los códigos de privacidad y las preferencias de ordenado.	<ul style="list-style-type: none"> <li>Miembro del club (negocio primario)</li> </ul>
Colocar una nueva orden	Este caso de uso describe el evento de un miembro del club que ingresa una orden para los productos de SoundStage.	<ul style="list-style-type: none"> <li>Miembro del club (negocio primario)</li> <li>Centro de distribución (receptor externo)</li> <li>Cuentas por pagar/cuentas por cobrar (servidor externo)</li> </ul>
Revisar la orden	Este caso de uso describe el evento de un miembro del club que revisa una orden colocada previamente. (La orden no debe haber sido mandada.)	<ul style="list-style-type: none"> <li>Miembro del club (negocio primario)</li> <li>Centro de distribución (receptor externo)</li> <li>Cuentas por pagar/cuentas por cobrar (servidor externo)</li> </ul>

**FIGURA 6.10** Conclusión

CANCELACIÓN DE LA ORDEN	Este caso de uso describe el evento de un miembro del club que cancela una orden previamente colocada. (La orden no debe haberse mandado.)	<ul style="list-style-type: none"> <li>• Miembro del club (negocio primario)</li> <li>• Centro de distribución (receptor externo)</li> <li>• Cuentas por pagar/cuentas por cobrar (servidor externo)</li> </ul>
HACER UNA INVESTIGACIÓN DE PRODUCTO	Este caso de uso describe el evento de un miembro del club que ve los productos para una posible compra. (Impulsado por el requerimiento de acceso a la Red.)	<ul style="list-style-type: none"> <li>• Miembro del club (negocio primario)</li> </ul>
HACER UNA INVESTIGACIÓN DEL HISTORIAL DE LA COMPRA	Este caso de uso describe el evento de un miembro del club que ve su historial de compra. (Tiempo límite de tres años.)	<ul style="list-style-type: none"> <li>• Miembro del club (negocio primario)</li> </ul>
ESTABLECER UN NUEVO PROGRAMA DE SUSCRIPCIÓN DE MIEMBROS	Este caso de uso describe el evento del departamento de mercadeo que establece un nuevo plan de suscripción de membresía para atraer nuevos miembros.	<ul style="list-style-type: none"> <li>• Mercadeo (negocio primario)</li> </ul>
INGRESAR LOS CAMBIOS DEL PROGRAMA DE SUSCRIPCIONES	Este caso de uso describe el evento del departamento de mercadeo que cambia el plan de suscripción para los miembros del club (por ejemplo, la extensión del periodo de cumplimiento).	<ul style="list-style-type: none"> <li>• Mercadeo (negocio primario)</li> </ul>
ESTABLECER UN PROGRAMA DE RESUSCRIPCIÓN DE EX MIEMBROS	Este caso de uso describe el evento del departamento de mercadeo que establece un plan de resuscripción para atraer nuevamente a los ex miembros.	<ul style="list-style-type: none"> <li>• Mercadeo (negocio primario)</li> </ul>
INGRESAR LOS CAMBIOS DEL PERFIL DE LOS MIEMBROS	Este caso de uso describe el evento del departamento de mercadeo que establece un nuevo plan de promoción para tentar a los miembros activos e inactivos para que ordenen el producto. (Nota: Una promoción se caracteriza por anuncios específicos, generalmente nuevos, de que la compañía está tratando de vender a un precio especial. Estas promociones se integran en un catálogo que se manda [o se comunica] a todos los miembros.)	<ul style="list-style-type: none"> <li>• Mercadeo (negocio primario)</li> </ul>
REVISAR LA PROMOCIÓN	Este caso de uso describe el evento del departamento de mercadeo que revisa una promoción.	<ul style="list-style-type: none"> <li>• Mercadeo (negocio primario)</li> </ul>
GENERAR EL REPORTE DIARIO DE CONCILIACIÓN DE 10-30-60 DÍAS POR OMISIÓN	Este caso de uso describe el evento de un reporte que se genera cotidianamente para listar a los miembros que no hayan cumplido con el acuerdo de comprar el número requerido de productos esbozados cuando se suscribieron. Este reporte está ordenado de acuerdo con el orden de los miembros que tienen 10 días de vencimiento, 30 días de vencimiento y 60 días de vencimiento.	<ul style="list-style-type: none"> <li>• Tiempo (actor iniciante)</li> <li>• Servicios a los miembros (receptor primario,* externo)</li> </ul>

\* Considerado primario porque recibe algo de valor medible.



**FIGURA 6.11** Diagrama del modelo de caso de uso del Sistema de servicios para los miembros de SoundStage

del sistema. El diagrama de casos de uso para los casos de uso listados en la figura 6.10 se muestra en la figura 6.11. Se creó usando el *System Architect* del Software Popkin y representa las relaciones entre los actores y los casos de uso. Además, los casos de uso han sido agrupados en subsistemas de negocios. Los subsistemas (símbolo de paquete de UML) representan las áreas funcionales lógicas de los procesos de negocios. La división del comportamiento del sistema en subsistemas es muy importante para entender la arquitectura del sistema y es clave para definir su estrategia de desarrollo; cuáles casos de uso serán desarrollados primero y por quién. Hemos rotulado las asociaciones entre los actores y los casos de uso como “iniciados” porque la herramienta no soportaba líneas con cabezas de flecha en ese momento. Tampoco incluimos a los actores proveedores y receptores externos debido a las limitaciones de espacio. La modelación de todos los casos de uso de un sistema específico puede requerir la creación de varios diagramas de modelos de casos de uso; como usted recuerda, un sistema puede contener docenas de casos de uso. En ese caso tal vez usted quiera crear un diagrama de modelo de casos de uso por separado para cada subsistema.

#### > Paso 4: Narraciones de los casos de uso para los requerimientos de documentos para los negocios

Cuando usted prepara las narraciones, es prudente documentarlas primero a un *alto nivel* para obtener rápidamente una comprensión de los eventos y de la magnitud del sistema. Entonces regrese a cada caso de uso y expándalo a una narración totalmente documentada de requerimientos de negocios. La figura 6.12 representa una narración de casos de uso de requerimientos para el caso de uso de Colocar una nueva orden en el Sistema de servicios

**Sistema de servicios para los miembros**Autor(es): 1Fecha: 2Versión: 3

Nombre del caso de uso:	Colocar nueva orden <u>4</u>	Caso de uso del tipo de requerimientos de negocios: <input checked="" type="checkbox"/> 5	
ID del caso de uso:	MSS-BUC002.00 <u>6</u>		
Prioridad:	Alta <u>7</u>		
Fuente:	Requerimiento: MSS-R1.00 <u>8</u>		
Actor primario de negocios:	Miembro del club <u>9</u>		
Otros actores participantes:	<ul style="list-style-type: none"> <li>• Almacén (receptor externo)</li> <li>• Cuentas por cobrar (servidor externo) <u>10</u></li> </ul>		
Otros involucrados interesados:	<u>11</u> <ul style="list-style-type: none"> <li>• Mercadeo: interesados en las actividades de ventas con objeto de planear nuevas promociones.</li> <li>• Suministro: interesados en las actividades de ventas con objeto de reponer el inventario.</li> <li>• Administración: interesados en la actividad de las órdenes con objeto de evaluar el desempeño de la compañía y la satisfacción del cliente (miembro).</li> </ul>		
Descripción:	<u>12</u> <p>Este caso de uso describe el evento de un miembro del club que ingresa una nueva orden de productos de SoundStage. Una vez que se verifica que los productos están en existencia se manda al almacén una orden de embalaje para que se prepare el flete. Para cualquier producto que no esté en existencia, se crea una orden de devolución. Al completarse, al miembro se le manda una confirmación de la orden.</p>		

**FIGURA 6.12** Narración del caso de uso Colocar una nueva orden en su versión de alto nivel

para los miembros. Observe que describe claramente el evento, que incluye los siguientes incisos:

- ① **Autor:** Los nombres de los individuos que contribuyeron a la escritura del caso de uso y que suministran un punto de contacto para cualquier persona que requiera información adicional acerca del caso de uso.
- ② **Fecha:** La fecha de la última modificación del caso de uso.
- ③ **Versión:** La versión actual del caso de uso (por ejemplo, 1.0).
- ④ **Nombre del caso de uso:** El nombre del caso de uso deberá representar la meta que el caso de uso está tratando de lograr. El nombre debe comenzar con un verbo (por ejemplo, Ingresar la Orden del nuevo miembro).
- ⑤ **Tipo de caso de uso:** Al realizar la modelación de los casos de uso se construyen primero los casos de uso de los requerimientos de negocios que se centren en la visión estratégica y en las metas de los diferentes involucrados. Este tipo de caso de uso está orientado hacia los negocios y refleja una vista de alto nivel del comportamiento deseado del sistema. Está libre de los detalles técnicos y puede incluir actividades manuales así como aquellas que se automatizarán. Los casos de uso de los requerimientos de negocios suministran una comprensión general del dominio y el alcance del problema pero no incluyen el detalle necesario para comunicar a los desarrolladores lo que el sistema debe hacer.
- ⑥ **ID del caso de uso:** Un identificador que distingue de manera única al caso de uso.
- ⑦ **Prioridad:** La prioridad comunica la importancia del caso de uso en términos de alta, mediana o baja.
- ⑧ **Fuente:** La fuente define a la entidad que originó la creación del caso de uso. Ésta podría ser un requerimiento, un documento específico, o un involucrado.
- ⑨ **Actor primario de negocios:** Es el involucrado que se beneficia en primer lugar del caso de uso al recibir algo de valor medible u observable.

- ⑩ *Otros actores participantes:* Otros actores que participan en el caso de uso para lograr su meta incluyen los actores que inician, los actores facilitadores, los actores servidores/receptores y los actores secundarios. Siempre incluya la manera en que el actor participa.
- ⑪ *Involucrado(s) interesado(s):* Un involucrado es cualquier persona que tenga un aporte en el desarrollo y la operación del sistema de software. Un involucrado que tenga interés es una persona (diferente de un actor) que tiene un interés creado en la meta del caso de uso.
- ⑫ *Descripción:* Una descripción corta resumida que consiste en un par de oraciones que esbozan el propósito del caso de uso y sus actividades.

**Documentación del curso de los eventos del caso de uso** Para cada caso de uso de alto nivel ya identificado, ahora debemos expandirlo para incluir el curso típico de los eventos del caso de uso y los cursos alternos. El curso típico de los eventos del caso de uso es una descripción paso por paso que comienza con el actor que inicia el caso de uso y que continúa hasta el final del evento del negocio. En esta sección incluimos solamente los pasos principales que ocurren la mayoría de las veces (su curso típico). El curso alterno documenta las excepciones o la bifurcación condicional del caso de uso. La figura 6.13 representa una narración de los requerimientos de un caso de uso para el caso de uso de Colocar una nueva orden del Sistema de Servicios para los Miembros. Observe que incluye los siguientes elementos adicionales:

- ① *Precondición:* Es una restricción del estado del sistema antes de la ejecución del caso de uso. En general, esto se refiere a otro caso de uso que debe ejecutarse previamente.
- ② *Disparador (generador):* Es el evento que inició la ejecución del caso de uso. Frequentemente, ésta es una acción física. Tal como un cliente que camina hasta el mostrador de ventas o un cheque que llega por el correo. El tiempo también puede disparar los casos de uso.
- ③ *Curso típico de los eventos:* Es la secuencia normal de las actividades realizadas por el(los) actor(es) y por el sistema con objeto de satisfacer la meta del caso de uso. Se incluyen las interacciones entre el sistema y el actor y las actividades realizadas por el sistema como respuesta a las interacciones. Observe que las acciones del actor se registran en la columna izquierda mientras que las acciones de los sistemas se registran en la columna derecha.
- ④ *Cursos alternos:* Los cursos alternos documentan los comportamientos del caso de uso si ocurre una excepción o una variación del curso típico. Esto puede suceder cuando ocurra un punto de decisión dentro del caso de uso o una excepción que requiera pasos adicionales fuera del alcance del curso típico.
- ⑤ *Conclusión:* Establece cuando termina con éxito el caso de uso; en otras palabras, cuando el actor primario recibe algo de valor medible.
- ⑥ *Postcondición:* Una postcondición es una restricción del estado del sistema después que el caso de uso ha sido ejecutado con éxito. Esto podría ser datos registrados en una base de datos o un recibo entregado a un cliente.
- ⑦ *Reglas de negocios:* Las reglas de negocios especifican políticas y procedimientos del negocio que el nuevo sistema debe obedecer. Esto podría incluir el cálculo de los cargos de embarque o las reglas para conceder plazos de crédito.
- ⑧ *Restricciones y especificaciones de la implantación:* Las restricciones y especificaciones de la implantación especifican requerimientos no funcionales que puedan impactar la realización del caso de uso y pueden ser de ayuda en cualquier planeación y alcance de la arquitectura. Los elementos que pueden incluirse son las especificaciones de seguridad, los requerimientos de la interfaz, etcétera.
- ⑨ *Hipótesis:* Cualesquiera hipótesis que formuló el creador al documentar el caso de uso.
- ⑩ *Aspectos abiertos:* Preguntas o aspectos que deben resolverse o investigarse antes de finalizar el caso de uso.

**Sistema de servicios para los miembros****Autor(es):** \_\_\_\_\_**Fecha:** \_\_\_\_\_**Versión:**

<b>Nombre del caso de uso:</b>	Colocar nueva orden	<b>Caso de uso del tipo de requerimientos de negocios:</b> <input checked="" type="checkbox"/>	
<b>ID del caso de uso:</b>	MSS-BUC002.00		
<b>Prioridad:</b>	Alta		
<b>Fuente:</b>	Requerimiento: MSS-R1.00		
<b>Actor primario de negocios:</b>	Miembro del club		
<b>Otros actores participantes:</b>	<ul style="list-style-type: none"> <li>• Almacén (receptor externo)</li> <li>• Cuentas por cobrar (servidor externo)</li> </ul>		
<b>Otros involucrados interesados:</b>	<ul style="list-style-type: none"> <li>• Mercadeo: interesados en las actividades de ventas con objeto de planear nuevas promociones.</li> <li>• Suministro: interesados en las actividades de ventas con objeto de reponer el inventario.</li> <li>• Administración: interesados en la actividad de las órdenes con objeto de evaluar el desempeño de la compañía y la satisfacción del cliente (miembro).</li> </ul>		
<b>Descripción:</b>	Este caso de uso describe el evento de un miembro del club que ingresa una nueva orden de productos de SoundStage. Una vez que se verifica que los productos están en existencia, se manda al almacén una orden de embalaje para que se prepare el flete. Para cualquier producto que no esté en existencia se crea una orden de devolución. Al completarse, al miembro se le manda una confirmación de la orden.		
<b>Precondición:</b> ①	La parte (persona o compañía) que ingresa la orden debe ser miembro.		
<b>Ocasionalor:</b> ②	Este caso de uso se inicia cuando se ingresa una nueva orden.		
<b>Curso típico de eventos:</b> ③	<b>Acción del actor</b> <b>Paso 1:</b> El miembro del club proporciona su información demográfica así como la información de las órdenes y de los pagos.	<b>Respuesta del sistema</b> <b>Paso 2:</b> El sistema responde verificando que se ha suministrado toda la información requerida. <b>Paso 3:</b> El sistema verifica la información demográfica del miembro del club contra lo que se ha registrado anteriormente. <b>Paso 4:</b> Para cada producto ordenado, el sistema valida la identidad del producto. <b>Paso 5:</b> Para cada producto ordenado, el sistema verifica la disponibilidad del producto. <b>Paso 6:</b> Para cada producto disponible, el sistema determina el precio que debe cobrarse al miembro del club. <b>Paso 7:</b> Una vez que se procesan todos los productos ordenados, el sistema determina el costo total de la orden. <b>Paso 8:</b> El sistema verifica el estado de la cuenta del miembro del club. <b>Paso 9:</b> El sistema valida el pago del miembro del club si existe. <b>Paso 10:</b> El sistema registra la información de la orden y luego libera la orden al centro de distribución apropiado (almacén) para llenarla. <b>Paso 11:</b> Una vez que se procesa la orden, el sistema genera una confirmación de la orden y la manda al miembro del club.	

**FIGURA 6.13** Versión expandida de la narración del caso de uso de Colocar una nueva orden

Los casos de uso de los requerimientos de negocios son herramientas excelentes porque describen los eventos que la organización debe procesar y responderles, pero les falta información con respecto a las interfaces y las actividades que están programadas para automatizarse por la tecnología de información. Posteriormente, en el capítulo 9, usted aprenderá cómo hacer evolucionar el caso de uso para incluir los detalles técnicos y de implantación.

<b>Cursos alternos:</b>	(4)	<p><b>Paso alternativo 2:</b> El miembro del club no ha suministrado toda la información necesaria para procesar la orden. Se notifica la discrepancia al miembro del club y se le urge a que vuelva a presentar la solicitud.</p> <p><b>Paso alternativo 3:</b> Si la información suministrada del miembro del club es diferente de lo que se registró anteriormente, verifique lo que está registrado actualmente, y luego actualice de acuerdo con esto la información del miembro del club.</p> <p><b>Paso alternativo 4:</b> Si la información de producto que suministró el miembro del club no concuerda con ninguno de los productos de SoundStage, notifique la discrepancia al miembro del club y solicite una aclaración.</p> <p><b>Paso alternativo 5:</b> Si no está disponible la cantidad ordenada del producto, se crea una orden de devolución.</p> <p><b>Paso alternativo 8:</b> Si el estado de la cuenta del miembro del club es que no tiene derechos, registre la información de la orden y póngala en estado de espera. Notifique el estado de la cuenta al miembro del club y la razón por la cual la orden está detenida. Finiquite el caso de uso.</p> <p><b>Paso alternativo 9:</b> Si el pago provisto por el miembro del club (tarjeta de crédito) no puede validarse, notifique al miembro del club y solicite un medio alterno de pago. Si el miembro del club no puede suministrar un medio alterno, cancele la orden y finiquite el caso de uso.</p>
<b>Conclusión:</b>	(5)	Este caso de uso concluye cuando el miembro del club recibe una confirmación de la orden.
<b>Postcondición:</b>	(6)	La orden ha sido registrada y si estaban disponibles los productos ordenados, éstos fueron liberados. Para cualquier producto no disponible se ha creado una orden de devolución.
<b>Reglas de negocios:</b>	(7)	<ul style="list-style-type: none"> <li>El miembro del club que responde a una promoción o un miembro que ejerce un crédito puede afectar el precio de cada artículo ordenado.</li> <li>Con las órdenes no se acepta efectivo ni cheques. Si llegan, serán regresados al miembro del club.</li> <li>Los productos se facturan al miembro del club solamente cuando han sido fletados.</li> </ul>
<b>Restricciones y especificaciones de implantación:</b>	(8)	<ul style="list-style-type: none"> <li>Debe suministrarse un GUI al socio de los Servicios para los miembros, y al miembro del club debe suministrarse una pantalla de la Red.</li> </ul>
<b>Hipótesis:</b>	(9)	La procuración de las órdenes de devolución será notificada mediante un reporte diario (caso de uso por separado).
<b>Aspectos abiertos:</b>	(10)	<ol style="list-style-type: none"> <li>Necesidad de determinar cómo se asignan los centros de distribución.</li> </ol>

**FIGURA 6.13** Conclusión

## Los casos de uso y la administración de proyectos

Como usted recordará, uno de los beneficios de la modelación de los casos de uso es que el modelo del caso de uso puede usarse para impulsar el esfuerzo completo de desarrollo de sistemas. Una vez que esté completo el modelo de caso de uso de los requerimientos de negocios, el administrador de proyectos o el analista de sistemas usa los casos de uso para los requerimientos de negocios para planear (estimar y programar) los ciclos de construcción del proyecto. Esto es especialmente crucial al aplicar el enfoque iterativo y de incrementos al desarrollo de software. A un ciclo de construcción, que consiste en las actividades del análisis, el diseño y la construcción de sistemas, se le asigna su alcance basándose en la importancia del caso de uso y en el tiempo que toma implantar el caso de uso. En otras palabras, se desarrollarán uno o más casos de uso para cada ciclo de construcción. Cuando un caso de uso es demasiado grande o complejo para terminarse en un ciclo de construcción, entonces inicialmente se implantará una versión simplificada, seguida por la versión completa en el siguiente ciclo de construcción. Para determinar la importancia de los casos de uso, el administrador de proyecto o el analista de sistemas terminarán la jerarquización de los casos de uso y la matriz de evaluación y construirá un diagrama de dependencia de casos de uso con entradas de los involucrados y del equipo de desarrollo. Usted aprenderá como usar estas herramientas en las siguientes secciones.

**matriz de jerarquía y prioridad de los casos de uso** Herramienta usada para evaluar los casos de uso y determinar su importancia.

### > Cómo jerarquizar y evaluar los casos de uso

En la mayoría de los proyectos de desarrollo de software, los casos de uso más importantes se desarrollan primero. Con objeto de determinar la prioridad de los casos de uso, el administrador de proyectos usa una herramienta llamada **matriz de jerarquía y priori-**

Nombre del caso de uso	Criterios de jerarquización, 1 a 5						Calificación total	Prioridad	Ciclo de construcción
	1	2	3	4	5	6			
Ingresar la orden de suscripción	5	5	5	4	5	5	29	Alto	1
Colocar la nueva orden	4	4	5	4	5	5	27	Alto	2
Hacer investigación de producto	1	1	1	1	1	1	6	Bajo	3
Establecer un programa de suscripción de nuevos miembros	4	5	5	3	5	5	27	Alto	1
Generar un reporte de conformidad por omisión diario de 10-30-60 días	1	1	1	1	1	1	6	Bajo	3
Revisar la orden	2	2	3	3	4	4	18	Medio	2

**FIGURA 6.14** Jerarquización y matriz de prioridad parciales de un caso de uso

**dad de los casos de uso.** Esta matriz se llena con las entradas de los involucrados y del equipo de desarrollo. Esta matriz, adaptada del trabajo de Craig Larman,<sup>3</sup> evalúa los casos de uso en una escala de 1 a 5 con respecto a seis criterios. Éstos son los siguientes:

1. Impacto significativo sobre el diseño de la arquitectura.
2. Fácil de implantar pero contiene una funcionalidad significativa.
3. Incluye funciones riesgosas, críticas con respecto al tiempo, o que son complejas.
4. Implica mucha investigación o tecnología nueva o de alto riesgo.
5. Incluye funciones primarias de negocios.
6. Aumentará las utilidades o disminuirá los costos.

Una vez que cada categoría ha sido calificada se anotan las calificaciones individuales, lo que conduce a la calificación final del caso de uso. A los casos de uso con las calificaciones más altas se les asignan la prioridad más alta y deben desarrollarse primero.

La figura 6.14 es una matriz parcial de prioridades y una jerarquización parcial de los casos de uso para el Sistema de servicios a los miembros. Basándose en los resultados del análisis, el caso de uso Ingresar la orden de suscripción debe desarrollarse primero. Pero no podemos estar seguros hasta que analicemos las dependencias del caso de uso.

### > Identificación de las dependencias de los casos de uso

Algunos casos de uso pueden depender de otros, con un caso de uso dejando al sistema en un estado que es una precondición para otro caso de uso. Por ejemplo, una precondición para mandar una promoción del club es que la promoción debe crearse primero. Usamos un diagrama llamado el **diagrama de dependencia de casos de uso** para modelar estas dependencias. El diagrama de dependencia de casos de uso proporciona los siguientes beneficios:

- La ilustración gráfica de los eventos del sistema y de sus estados aumenta la comprensión de la funcionalidad del sistema.
- Ayuda a identificar los casos de uso faltantes. Un caso de uso con una precondición que no se satisface con la ejecución de cualquier otro caso de uso puede indicar un caso de uso faltante.
- Ayuda a facilitar la administración del proyecto al ilustrar cuáles son los casos de uso más críticos (que tienen el mayor número de dependencias) y entonces deben tener una prioridad más alta.

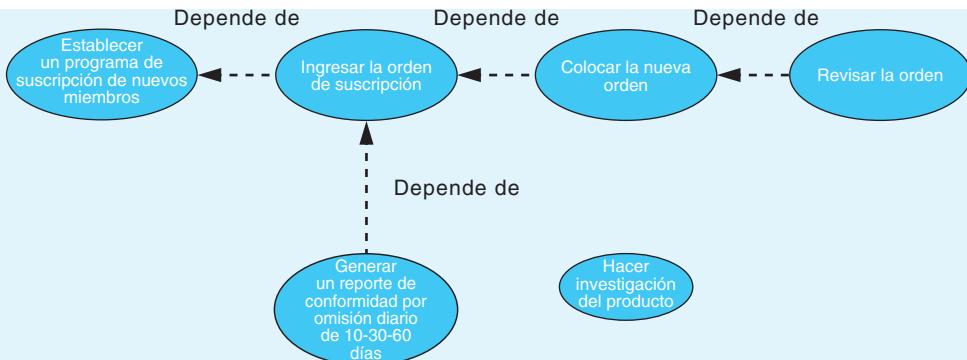
**diagrama de dependencia de casos de uso** Representación gráfica de las dependencias entre casos de uso.

La figura 6.15 es el diagrama de dependencia de casos de uso para aquéllos listados en la figura 6.14. Los casos de uso que son dependientes entre sí se conectan con una

<sup>3</sup> Craig Larman, *Applying UML Patterns* (Upper Saddle River, NJ: Prentice Hall, 1998).

**FIGURA 6.15**

Ejemplo de un diagrama de dependencia de un caso de uso



línea segmentada rotulada “Depende de”. En la figura 6.15, el caso de uso Ingresar la orden de suscripción tiene una dependencia (precondición) con respecto al caso de uso Establecer un nuevo programa de suscripción de miembros. Debido a esta dependencia, el caso de uso Establecer un nuevo programa de suscripción de miembros debe desarrollarse primero aun cuando Ingresar la orden de suscripción tenga una calificación más alta como se refleja en la figura 6.14.

## Mapa de aprendizaje

Este capítulo presentó una introducción a los casos de uso y cómo pueden usarse para documentar los requerimientos funcionales. Usted también ha aprendido que la modelación de casos de uso basada en los conceptos orientados a objetos es una herramienta complementaria excelente para las herramientas tradicionales del análisis y diseño de sistemas tales como la modelación de los procesos y la modelación de los datos. Muchos de ustedes proseguirán directamente al capítulo 7, “Modelado y análisis de datos”. Todos los sistemas de información incluyen bases de datos, y la modelación de datos es una habilidad esencial para el desarrollo de las bases de datos. También es más fácil sincronizar los modelos de datos con los modelos de procesos que al revés. Su profesor podría preferir que usted estudie primero el capítulo 8, “Modelado de procesos”. La modelación de procesos es una forma efectiva de analizar y documentar los requerimientos funcionales del sistema. En los cursos que siguen un enfoque orientado a objetos se recomienda revisar algún libro sobre análisis y modelación orientados a objetos con el uso de lenguaje unificado de modelación (UML).

## Resumen



- Hay dos elementos primarios que participan al realizar la modelación de los casos de uso. El primero es el diagrama de caso de uso, que ilustra gráficamente al sistema como una colección de casos de uso, actores (usuarios) y sus relaciones. Los detalles de cada evento de negocios y de cómo interactúan los usuarios con el sistema se describen en el segundo elemento, llamado la narración del caso de uso, que es la descripción textual del evento de

- negocios y cómo va a interactuar el usuario con el sistema para lograr la tarea.
- La modelación de casos de uso utiliza dos construcciones: los actores y los casos de uso. Un actor representa cualquier cosa que necesite interaccionar con el sistema para intercambiar información. Un actor es un usuario, un papel, que podría ser un sistema externo así como una persona. Un caso de uso es una secuencia de pasos relacionados

(un escenario) tanto automatizada como manual, con el propósito de completar una tarea individual del negocio.

3. Hay principalmente cuatro tipos de actores:

- a) *Actor primario de negocios*: El involucrado que se beneficia en primer lugar de la ejecución del caso de uso al recibir algo de valor medible u observable.
- b) *Actor primario del sistema*: El involucrado que sostiene una interfaz directa con el sistema para iniciar u originar el evento de negocios o del sistema.
- c) *Actor externo proveedor*: El involucrado que responde a una solicitud proveniente del caso de uso.
- d) *Actor externo receptor*: El involucrado que no es el actor primario pero que recibe algo de valor medible u observable (salida) proveniente del caso de uso.

4. Los eventos temporales son eventos de negocios que se realizan (o se originan) en forma automática; cuando se cumple cierta fecha o lapso. Debido a eso, decimos que el actor de un evento temporal es el tiempo.

5. Una relación se ilustra como una línea entre dos símbolos en el diagrama de casos de uso.

- a) Una asociación es una relación entre un actor y un caso de uso.

b) La relación entre el caso de uso de extensión y el caso de uso que se está extendiendo se llama una relación de extensión.

c) La relación entre el caso de uso resumen y el caso de uso que los usa se llama una relación de usos.

d) La relación de herencia ocurre cuando un actor hereda la capacidad para iniciar un caso de uso de otro.

e) La relación de dependencia indica una dependencia entre los casos de uso. En otras palabras, la precondición de un caso de uso depende de la postcondición de otro caso de uso.

6. Los pasos requeridos para producir un modelo de casos de uso de requerimientos son los siguientes:

- a) Identificar a los actores de negocios.
- b) Identificar los casos de uso de requerimientos de negocios.
- c) Construir el diagrama del modelo de casos de uso.
- d) Documentar las narraciones de los casos de uso para los requerimientos de negocios.

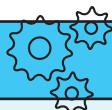
7. La matriz de prioridad y la jerarquización del caso de uso son las herramientas que usan los administradores de proyecto para priorizar y programar el desarrollo del caso de uso.

## Preguntas de repaso

1. ¿Qué es el desarrollo centrado en el usuario y por qué es crítico para el éxito del proceso de desarrollo de sistema?
2. ¿Cómo se relaciona la modelación de los casos de uso con el desarrollo centrado en el usuario?
3. Además de alentar la participación del usuario, la modelación de los casos de uso proporciona muchos otros beneficios; lístelos.
4. El modelado de caso de uso emplea dos elementos primarios: el diagrama de caso de uso y la narración de caso de uso. ¿Cómo se utilizan estos dos elementos y cuáles son sus diferencias?
5. Los diagramas de caso de uso constan de tres componentes. ¿Cuáles son estos tres componentes y cuál es su propósito?
6. ¿Cómo se usan los casos de uso durante el ciclo de vida completo del desarrollo del sistema?
7. De las cuatro categorías primarias de actores, ¿quién es el actor primario del sistema?

8. ¿Cuáles son los tipos diferentes de relaciones empleadas en un diagrama de caso de uso y cuál es su propósito?
9. ¿Cuál es el objetivo de construir el modelo de casos de uso de los requerimientos y qué pasos deben seguirse?
10. ¿Por qué es la identificación de los actores el primer paso en la modelación de los casos de uso?
11. ¿De qué deberíamos darnos cuenta cuando buscamos casos de uso de requerimientos de negocios?
12. ¿Cuál es el curso de eventos típico de un caso de uso?
13. ¿Por qué son esenciales la jerarquización y la evaluación de los casos de uso?
14. ¿Cuáles son los seis criterios en la jerarquización y en la matriz prioritaria?
15. ¿Cuál es el diagrama de dependencia de los casos de uso y por qué lo usamos?

## Problemas y ejercicios



- De acuerdo con el autor Fred Brooks, ¿qué es lo más difícil de hacer en el desarrollo de sistemas? ¿Cómo ayuda la modelación de casos de uso en esta área?
- En la modelación de casos de uso, ¿cuáles son los dos elementos principales que usa el analista de sistemas? Describa cada uno de estos elementos y explique su propósito.
- ¿Qué debe tener siempre en mente un analista de sistemas al identificar y desarrollar los casos de uso en relación con su propósito? Como la exploración de requerimientos se completó previamente, ¿es en realidad necesario pasar mucho tiempo con los usuarios en este punto? ¿Simplemente qué debería representar un caso de uso? ¿Un caso de uso equivale a un requerimiento funcional?
- ¿Durante qué parte del desarrollo del ciclo de vida de se definen por primera vez los casos de uso? ¿Cuándo se usan durante el ciclo de vida de desarrollo, y para qué propósito?
- Relacione a los siguientes involucrados y usuarios externos con el actor correcto. ¿Qué es un evento temporal? ¿Quién o qué se considera como el actor en un evento temporal, y por qué?

### Los involucrados y los usuarios externos

- Servicio Postal de Estados Unidos
- Cerrojo con teclado
- Agente de autos de alquiler
- Gerente de ventas que genera el informe regional de ventas
- Administrador de ventas que recibe el informe regional de ventas
- Sistema automático de aspersor giratorio para regar el césped
- El conductor que compra gasolina con tarjeta de cajero automático
- El servicio de la autorización de préstamo bancario

- ¿Cuál es el tipo de relación para cada uno de los siguientes ejemplos?

- La relación entre el caso de uso "Imprimir la forma" y varios otros casos de uso que implican la impresión de tipos diferentes de formas.
- La relación entre un oficial de motocicleta y un dispositivo de mano para escribir citatorios.
- La relación entre un cliente y un dependiente vendedor de una tienda que pueden interrogar

al sistema de inventario para ver si un artículo existe en inventario, y un actor creado específicamente para minimizar la comunicación duplicada de sistema.

- La relación entre el caso de uso "Calcular el promedio general" y el caso de uso prolongado "Crear un certificado de estudios".
- La relación entre el caso de uso "Enviar orden" y el caso de uso "Colocar una orden".
- Y&J Cookbooks es un negocio pequeño ficticio poseído y manejado por una pareja jubilada. Hasta este momento, Y&J Cookbooks han vendido sus libros sólo a través de pedidos por correo. Los dueños ahora quieren desarrollar un sistema en línea para vender libros de cocina raros y agotados a través de Internet. Los visitantes podrán hojear diferentes libros de cocina, pero tendrán que crear una cuenta del cliente antes de poder hacer una compra. Los pagos se aceptarán sólo en línea con una tarjeta de crédito reconocida, y se verificará la tarjeta de crédito antes de que la orden pueda ser aprobada. Basado en esta información, identifique a los actores principales del negocio.
- En la modelación de casos de uso, una vez que usted identifica a los actores de negocios, ¿qué perspectiva y qué lenguaje debería usar usted para definirlos? Use esa perspectiva y ese lenguaje para construir un glosario de actores usando la figura 6.8 como ejemplo.
- Un diagrama de contexto puede ayudar muchísimo para identificar los diferentes casos de uso. Prepare un diagrama de contexto de alto nivel para el sitio Web de Y&J Cookbooks, usando la figura 6.9 como ejemplo.
- El siguiente paso en la modelación de casos de uso para requerimientos es identificar los casos de uso de requerimientos de negocios. ¿Qué debería capturar cada caso de uso? ¿Qué técnica efectiva puede usted usar para identificar los casos de uso? ¿Qué preguntas puede hacer usted para identificar los casos de uso? ¿Cuál es la diferencia entre un caso de uso y un caso esencial de uso?
- El tercer paso en la modelación de casos de uso es construir los diagramas de modelo de caso de uso. Basado en el glosario de actores Y&J Cookbooks y en el diagrama de contexto, cree un diagrama de modelo de caso de uso de alto nivel, mostrando las interacciones entre el actor comprador/cliente y el sistema, incluyendo la búsqueda y el hojeado de libros, la compra y la operación de la cuenta del cliente. Asegúrese de demostrar las relaciones entre el actor y cada uno de estos casos de uso.
- El siguiente paso es crear narraciones de caso de uso para documentar los requerimientos de negocios. ¿Por qué la preparación de las narraciones generalmente se hace en un proceso de dos pasos y

- cuáles son estos dos pasos? Basado en el diagrama de modelo de casos de uso de alto nivel precedente, cree una narración expandida, usando la figura 6.13 como ejemplo.
13. ¿Cuál es la relación de la modelación de los casos de uso con la gerencia del proyecto? ¿Por qué es importante? ¿Por qué se jerarquizan los casos y qué

herramienta se usa para jerarquizarlos? ¿Quién proporciona la entrada para jerarquizarlos? ¿Qué criterios se usan para la jerarquización? Explique por qué es necesario identificar las dependencias de los casos de uso y suministre un ejemplo. ¿Qué herramienta se usa para identificar las dependencias?



## Proyectos e investigación

1. Al principio del capítulo 6 hay una cita tomada de un artículo por Frederick P. Brooks Jr., quien es generalmente considerado como uno de los contribuyentes y autores más destacados del campo del desarrollo de administración de proyecto y de software. Navegue en la Web para encontrar este artículo y otros artículos de Fred Brooks, y acerca de él.
  - a) Al hacer su búsqueda, ¿cuántas referencias encontró usted acerca del autor?
  - b) Basado en la información presentada en los capítulos previos, explique la declaración de Brook que “la parte más difícil en la construcción de un sistema de software es decidir precisamente qué construir”.
  - c) ¿Cómo se llama el artículo en el cual Brooks hizo la declaración precedente, y cuál fue el tema principal del artículo?
  - d) ¿Cuál es el libro de Brook mejor conocido que está todavía en prensa y ampliamente leído décadas después de su publicación original? ¿Cuál fue el tema principal de este libro?
  - e) ¿Cuál considera usted que es la máxima contribución de Brooks hasta la fecha? ¿Por qué?
2. El Standish Group, que se mencionó en el capítulo 6, es un grupo de investigación independiente que estudia cambios y tendencias en la tecnología de la información. En 1994, el Standish Group publicó su Reporte CHAOS de avanzada, el cual “expone [expuso] el fracaso abrumador de proyectos de desarrollo de aplicaciones de tecnología de la información en el ambiente actual de sistemas de información para la administración”. Desde entonces, el Standish Group ha publicado actualizaciones periódicas de su informe original. Visite su sitio Web en [www.standishgroup.com](http://www.standishgroup.com) y siga los enlaces en su área de acceso al público, donde usted puede encontrar un resumen de su último informe de investigación de CHAOS, así como también el informe original de 1994.
  - a) ¿Qué criterios usa el Standish Group para determinar si un proyecto tuvo éxito, fue cuestionado, o fracasó?
  - b) ¿Basado en el último informe de investigación, cuál fue el porcentaje de proyectos que tuvieron éxito, fueron cuestionados, o fracasaron?
3. Seleccione un sistema de información usado en su organización o en su escuela. Entreviste a un analista de sistemas o un diseñador que esté familiarizado con el sistema. Basado en la información prevista, haga lo siguiente:
  - a) Describa el sistema de información y de organización que usted seleccionó.
  - b) Realice un diagrama de contexto del sistema.
  - c) Identifique a los actores de negocios.
  - d) Escriba un glosario de actores.
  - e) Identifique los casos de uso esenciales de requerimientos de negocios.
  - f) Escriba un glosario de casos de uso.
4. Basado en la información prevista referente al sistema que usted seleccionó en la pregunta precedente:
  - a) Construya un diagrama de modelo de caso de uso que incluya todos los subsistemas principales.
  - b) Prepare narraciones expandidas de caso de uso para tres de los casos de uso esenciales.
  - c) Prepare la jerarquización y la matriz prioritaria del caso de uso, luego úsela para jerarquizar y evaluar los casos de uso.
  - d) Identifique las dependencias del caso de uso.
  - e) Prepare un diagrama de dependencia del caso de uso.

## Especificaciones de requerimientos con casos de uso

5. Navegue en la Web o busque en las publicaciones profesionales en su biblioteca de la escuela, artículos de investigación sobre los desarrollos nuevos y emergentes en la modelación de casos de uso. Seleccione dos artículos, luego haga lo siguiente:
  - a) Proporcione la referencia bibliográfica de cada artículo. (Emplee el formato usado por su escuela.)
  - b) Elabore un resumen con sus propias palabras para cada artículo.
  - c) Compare y contraste las metodologías descritas en cada artículo. Describa cuál piensa usted que es más viable, significativo o ambos, y explique por qué.
6. Realice entrevistas con varios desarrolladores con respecto a sus puntos de vista sobre la modelación de los casos de uso. Si es posible, trate de encontrar desarrolladores de organizaciones diferentes o que tengan grados diferentes de experiencia o ambas cosas, así como también tipos diferentes de experiencia (es decir, un desarrollador que tenga experiencia principalmente como analista de sistemas, uno con experiencia como diseñador, y otro como constructor).
  - a) Describa a los desarrolladores que usted entrevistó en términos de su experiencia. Por ejemplo, ¿cuánto tiempo han trabajado en la tecnología de la información, cuál es su área de especialización, y qué tan familiarizados están con la modelación de casos de uso?
  - b) ¿Cuál es la naturaleza de su(s) organización(es)?
  - c) ¿Qué preguntas hizo usted?
  - d) ¿Cuáles son los puntos de vista de cada desarrollador con respecto a la importancia y el valor de la modelación de los casos de uso?
  - e) ¿Utilizan estos desarrolladores realmente la modelación de casos de uso en su organización actual? ¿Por qué sí o por qué no?
  - f) Si ellos fueran el CIO de su organización por un día, ¿cambiarían la arquitectura de tecnología de la información de su organización con respecto a la modelación de los casos de uso? Si es así, ¿cómo?
  - g) ¿Usando el modelo de madurez de capacidad, cómo evaluaría usted el nivel de madurez de su organización? ¿Por qué?
  - b) ¿Qué conclusiones (si las hay) puede obtener usted de las entrevistas estimando la aplicación práctica de la modelación de los casos de uso?
  - i) ¿Cuáles fueron sus puntos de vista con respecto a la importancia y el valor de la modelación de casos de uso antes de las entrevistas? ¿Cambiaron sus puntos de vista como resultado de las entrevistas? Si es así, ¿cómo cambiaron y por qué?

## Casos breves



1. En un caso breve del capítulo 5, usted entrevistó a los involucrados con respecto a un sistema de inscripciones en línea. En ese ejercicio, usted tenía que comprender los asuntos y necesidades que los involucrados tuvieran en cuanto al sistema. Revise sus conclusiones de esas entrevistas.
  - a) Revise otros sistemas de inscripción escolar.  
¿Hay algún aspecto que usted piense que sería del agrado o desagrado de los involucrados? Tome notas y esboce ejemplos de pantallas de otros sistemas.
  - b) Realice una entrevista de seguimiento con los involucrados con quienes usted previamente

habló y determine la funcionalidad específica y los requerimientos de facilidad de uso para su escuela.

2. Elabore una descripción de caso de uso para al menos uno de los requerimientos de funcionalidad que usted encontró en el problema previo. Siga el ejemplo mostrado en la figura 6.10.
3. Identifique a todos los actores del sistema de inscripción de la escuela. ¿Cuáles son los casos de uso que iniciará cada uno de ellos?
4. Usando su respuesta para el problema previo, dibuje un diagrama de caso de uso del sistema de inscripción de la escuela.

## Ejercicios de equipo e individuales



1. Mesa redonda: ¿Piensa usted que las personas son siempre absolutamente veraces en sus respuestas a las preguntas de las entrevistas?
2. Observe una película muda. Mesa redonda: ¿Cuál es la comunicación que tiene lugar además de las palabras?
3. Mesa redonda: Cuando usted determina los requerimientos de un sistema a través de un método

tal como una entrevista, supone que la persona a quien entrevista y de quien recopila información, *quiere que el sistema tenga éxito*. ¿Es éste siempre el caso? ¿Cómo puede manejar usted la recopilación de requerimientos si no es ése el caso?



## Lecturas recomendadas

Ambler, Scott W. *The Object Primer*. Nueva York: Cambridge University Press, 2001. Información muy buena acerca de la documentación de los casos de uso y su uso.

Armour, Frank y Granville Miller. *Advance Use Case Modeling*. Boston: Addison-Wesley, 2001. Este libro presenta una excelente cobertura del proceso de la modelación de los casos de uso.

Brooks Jr., F.P., 1987. "No SilverBullet-Essence and Accidents of Software Engineering." *Computer* 20(4), 10-19 de abril. *proc. IFIP Congress*, Dublín, Irlanda, 1986.

Jacobson, Ivar; Magnus Christerson; Patrik Jonsson; y Gunnar Overgaard. *Object-Oriented Software Engineering—A Use Case Driven Approach*. Workingham, Inglaterra: Addison-Wesley, 1992. Este libro presenta una cobertura detallada de cómo identificar y documentar los casos de uso.

Larman, Craig. *Applying UML and Patterns*. Upper Saddle River, NJ: Prentice Hall, 1998. Este libro proporciona una vista panorámica comprensiva del proceso de la modelación de los casos de uso.



# Tema 7

Modelado de los requerimientos



# 7 Modelado de los requerimientos

En el nivel técnico, la ingeniería de software comienza con una serie de tareas de modelado que conducen a la especificación de los requerimientos y a la representación de un diseño del software que se va a elaborar. El modelo de requerimientos<sup>1</sup> —un conjunto de modelos, en realidad— es la primera representación técnica de un sistema.

En un libro fundamental sobre métodos para modelar los requerimientos, Tom DeMarco [DeM79] describe el proceso de la manera siguiente:

Al mirar retrospectivamente los problemas y las fallas detectados en la fase de análisis, concluyo que es necesario agregar lo siguiente al conjunto de objetivos de dicha fase. Debe ser muy fácil dar mantenimiento a los productos del análisis. Esto se aplica en particular al Documento de Objetivos [especificación de los requerimientos del software]. Los problemas grandes deben ser enfrentados con el empleo de un método eficaz para dividirlos. La especificación victoriana original resulta caduca. Deben usarse gráficas, siempre que sea posible. Es necesario diferenciar las consideraciones lógicas [esenciales] y las físicas [implementación]... Finalmente, se necesita... algo que ayude a dividir los requerimientos y a documentar dicha partición antes de elaborar la especificación... algunos medios para dar seguimiento a las interfaces y evaluarlas... nuevas herramientas para describir la lógica y la política, algo mejor que un texto narrativo.

## Una mirada rápida

*:Qué es?* La palabra escrita es un vehículo maravilloso para la comunicación, pero no necesariamente es la mejor forma de representar los requerimientos de software de computadora. El modelado de los requerimientos utiliza una combinación de texto y diagramas para ilustrarlos en forma que sea relativamente fácil de entender y, más importante, de revisar para corregir, completar y hacer congruente.

*:Quién lo hace?* Un ingeniero de software (a veces llamado “analista”) construye el modelo con el uso de los requerimientos recabados del cliente.

## CONCEPTOS CLAVE

análisis del dominio . . . . .	129
análisis gramatical . . . . .	143
asociaciones . . . . .	152
casos de uso . . . . .	132
clases de análisis . . . . .	143
diagrama de actividades .	137
diagrama de canal . . . . .	138
modelado basado en clases . . . . .	142
modelado basado en escenarios . . . . .	131
modelado CRC . . . . .	148
modelado de datos . . . . .	139
modelado de requerimientos . . . . .	130
modelos UML . . . . .	137
paquetes de análisis . . . . .	154

*¿Por qué es importante?* Para validar los requerimientos del software se necesita estudiarlos desde varios puntos de vista diferentes. En este capítulo se considerará el modelado de los requerimientos desde tres perspectivas distintas: modelos basados en el escenario, modelos de datos (información) y modelos basados en la clase. Cada una representa a los requerimientos en una “dimensión” diferente, con lo que aumenta la probabilidad de detectar errores, de que afloren las inconsistencias y de que se revelen las omisiones.

*¿Cuáles son los pasos?* El modelado basado en escenarios es una representación del sistema desde el punto de vista del usuario. El modelado basado en datos recrea el espacio de información e ilustra los objetos de datos que manipulará el software y las relaciones entre ellos. El modelado orientado a clases define objetos, atributos y relaciones. Una vez que se crean los modelos preliminares, se mejoran y analizan para evaluar si están claros y completos, y si son consistentes. En el capítulo 7 se amplían con representaciones adicionales las dimensiones del modelado descritas aquí, lo que da un punto de vista más sólido de los requerimientos.

*¿Cuál es el producto final?* Para construir el modelo de requerimientos, se escoge una amplia variedad de representaciones basadas en texto y en diagramas. Cada una de dichas representaciones da una perspectiva de uno o más de los elementos del modelo.

*¿Cómo me aseguro de que lo hice bien?* Los productos del trabajo para modelar los requerimientos deben revisarse para saber si son correctos, completos y consistentes. Deben reflejar las necesidades de todos los participantes y establecer el fundamento desde el que se realizará el diseño.

---

1 En ediciones anteriores de este libro, se usó el término *modelo de análisis*, en lugar de *modelo de requerimientos*. En esta edición, el autor decidió usar ambas expresiones para designar la actividad que define distintos aspectos del problema por resolver. *Análisis* es lo que ocurre cuando se obtienen los *requerimientos*.

Aunque DeMarco escribió hace más de un cuarto de siglo acerca de los atributos del modelado del análisis, sus comentarios aún son aplicables a los métodos y notación modernos del modelado de los requerimientos.

## 6.1 Análisis de los requerimientos

El análisis de los requerimientos da como resultado la especificación de las características operativas del software, indica la interfaz de éste y otros elementos del sistema, y establece las restricciones que limitan al software. El análisis de los requerimientos permite al profesional (sin importar si se llama *ingeniero de software, analista o modelista*) construir sobre los requerimientos básicos establecidos durante las tareas de concepción, indagación y negociación, que son parte de la ingeniería de los requerimientos (véase el capítulo 5).

La acción de modelar los requerimientos da como resultado uno o más de los siguientes tipos de modelo:

- *Modelos basados en el escenario* de los requerimientos desde el punto de vista de distintos “actores” del sistema.
- *Modelos de datos*, que ilustran el dominio de información del problema.
- *Modelos orientados a clases*, que representan clases orientadas a objetos (atributos y operaciones) y la manera en la que las clases colaboran para cumplir con los requerimientos del sistema.
- *Modelos orientados al flujo*, que representan los elementos funcionales del sistema y la manera como transforman los datos a medida que se avanza a través del sistema.
- *Modelos de comportamiento*, que ilustran el modo en el que se comparte el software como consecuencia de “eventos” externos.

**cita:** “Cualquier ‘vista’ de los requerimientos es insuficiente para entender o describir el comportamiento deseado de un sistema complejo.”

**Alan M. Davis**

Estos modelos dan al diseñador del software la información que se traduce en diseños de arquitectura, interfaz y componentes. Por último, el modelo de requerimientos (y la especificación de requerimientos de software) brinda al desarrollador y al cliente los medios para evaluar la calidad una vez construido el software.

Este capítulo se centra en el *modelado basado en escenarios*, técnica que cada vez es más popular entre la comunidad de la ingeniería de software; el *modelado basado en datos*, más especializado, apropiado en particular cuando debe crearse una aplicación o bien manipular un espacio complejo de información; y el *modelado orientado a clases*, representación de las clases orientada a objetos y a las colaboraciones resultantes que permiten que funcione el sistema. En el capítulo 7 se analizan los modelos orientados al flujo, al comportamiento, basados en el patrón y en *webapps*.

**punto clave** El modelo de análisis y la especificación de requerimientos proporcionan un medio para evaluar la calidad una vez construido el software.

### > 6.1.1 Objetivos y filosofía general

Durante el modelado de los requerimientos, la atención se centra en *qué*, no en *cómo*. ¿Qué interacción del usuario ocurre en una circunstancia particular?, ¿qué objetos manipula el sistema?, ¿qué funciones debe realizar el sistema?, ¿qué comportamientos tiene el sistema?, ¿qué interfaces se definen? y ¿qué restricciones son aplicables?<sup>2</sup>

**cita:** “Los requerimientos no son arquitectura. No son diseño ni la interfaz de usuario. Los requerimientos son las necesidades.”

**Andrew Hunt y David Thomas**

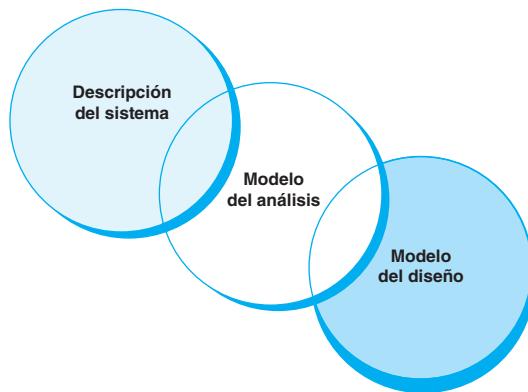
En los capítulos anteriores se dijo que en esta etapa tal vez no fuera posible tener la especificación completa de los requerimientos. El cliente quizás no esté seguro de qué es lo que requiere con precisión para ciertos aspectos del sistema. Puede ser que el desarrollador esté inseguro de que algún enfoque específico cumpla de manera apropiada la función y el desempeño. Estas realidades hablan a favor de un enfoque iterativo para el análisis y el modelado de los requerimientos. El analista debe modelar lo que se sabe y usar el modelo como base para el diseño del incremento del software.<sup>3</sup>

2 Debe notarse que, a medida que los clientes tienen más conocimientos tecnológicos, hay una tendencia hacia la especificación del *cómo* tanto como del *qué*. Sin embargo, la atención debe centrarse en el *qué*.

3 En un esfuerzo por entender mejor los requerimientos para el sistema, el equipo del software tiene la alternativa de escoger la creación de un prototipo (véase el capítulo 2).

**punto clave** El modelo de análisis debe describir lo que quiere el cliente, establecer una base para el diseño y un objetivo para la validación.

El modelo de requerimientos debe lograr tres objetivos principales: 1) describir lo que requiere el cliente, 2) establecer una base para la creación de un diseño de software y 3) definir un conjunto de requerimientos que puedan validarse una vez construido el software. El modelo de análisis es un puente entre la descripción en el nivel del sistema que se centra en éste en lo general o en la funcionalidad del negocio que se logra con la aplicación de software, hardware, datos, personas y otros elementos del sistema y un diseño de software (véanse los capítulos 8 a 13) que describa la arquitectura de la aplicación del software, la interfaz del usuario y la estructura en el nivel del componente. Esta relación se ilustra en la figura 6.1.



**FIGURA 6.1** El modelo de requerimientos como puente entre la descripción del sistema y el modelo del diseño.

Es importante observar que todos los elementos del modelo de requerimientos pueden rastrearse directamente hasta las partes del modelo del diseño. No siempre es posible la división clara entre las tareas del análisis y las del diseño en estas dos importantes actividades del modelado. Invariablemente, ocurre algo de diseño como parte del análisis y algo de análisis se lleva a cabo durante el diseño.

### > 6.1.2 Reglas prácticas del análisis

Arlow y Neustadt [Arl02] sugieren cierto número de reglas prácticas útiles que deben seguirse cuando se crea el modelo del análisis:

¿Hay lineamientos básicos que nos ayuden a hacer el trabajo de análisis de los requerimientos?

**cita:** “Los problemas que es benéfico atacar demuestran su beneficio con un contragolpe.”

Piet Hein

- *El modelo debe centrarse en los requerimientos que sean visibles dentro del problema o dentro del dominio del negocio. El nivel de abstracción debe ser relativamente elevado. “No se empantane en los detalles”* [Arl02] que traten de explicar cómo funciona el sistema.
- *Cada elemento del modelo de requerimientos debe agregarse al entendimiento general de los requerimientos del software y dar una visión del dominio de la información, de la función y del comportamiento del sistema.*
- *Hay que retrasar las consideraciones de la infraestructura y otros modelos no funcionales hasta llegar a la etapa del diseño.* Es decir, quizás se requiera una base de datos, pero las clases necesarias para implementarla, las funciones requeridas para acceder a ella y el comportamiento que tendrá cuando se use sólo deben considerarse después de que se haya terminado el análisis del dominio del problema.
- *Debe minimizarse el acoplamiento a través del sistema.* Es importante representar las relaciones entre las clases y funciones. Sin embargo, si el nivel de “interconectividad” es extremadamente alto, deben hacerse esfuerzos para reducirlo.
- *Es seguro que el modelo de requerimientos agrega valor para todos los participantes.* Cada actor tiene su propio uso para el modelo. Por ejemplo, los participantes de negocios deben usar el modelo para validar los requerimientos; los diseñadores deben usarlo como pase para el diseño; el personal de aseguramiento de la calidad lo debe emplear como ayuda para planear las pruebas de aceptación.

- Mantener el modelo tan sencillo como se pueda. No genere diagramas adicionales si no agregan nueva información. No utilice notación compleja si basta una sencilla lista.

### > 6.1.3 Análisis del dominio

Al estudiar la ingeniería de requerimientos (en el capítulo 5), se dijo que es frecuente que haya patrones de análisis que se repiten en muchas aplicaciones dentro de un dominio de negocio específico. Si éstos se definen y clasifican en forma tal que puedan reconocerse y aplicarse para resolver problemas comunes, la creación del modelo del análisis es más expedita. Más importante aún es que la probabilidad de aplicar patrones de diseño y componentes de software ejecutable se incrementa mucho. Esto mejora el tiempo para llegar al mercado y reduce los costos de desarrollo.

Pero, ¿cómo se reconocen por primera vez los patrones de análisis y clases? ¿Quién los define, clasifica y prepara para usarlos en los proyectos posteriores? La respuesta a estas preguntas está en el *análisis del dominio*. Firesmith [Fir93] lo describe del siguiente modo:

El análisis del dominio del software es la identificación, análisis y especificación de los requerimientos comunes, a partir de un dominio de aplicación específica, normalmente para usarlo varias veces en múltiples proyectos dentro del dominio de la aplicación [...] [El análisis del dominio orientado a objetos es] la identificación, análisis y especificación de capacidades comunes y reutilizables dentro de un dominio de aplicación específica en términos de objetos, clases, subensambles y estructuras comunes.

El “dominio de aplicación específica” se extiende desde el control electrónico de aviones hasta la banca, de los juegos de video en multimedios al software incrustado en equipos médicos. La meta del análisis del dominio es clara: encontrar o crear aquellas clases o patrones de análisis que sean aplicables en lo general, de modo que puedan volverse a usar.<sup>4</sup>

Con el empleo de la terminología que se introdujo antes en este libro, el análisis del dominio puede considerarse como una actividad sombrilla para el proceso del software. Esto significa que el análisis del dominio es una actividad de la ingeniería de software que no está conectada con ningún proyecto de software. En cierta forma, el papel del analista del dominio es similar al de un maestro herrero en un ambiente de manufactura pesada. El trabajo del herrero es diseñar y fabricar herramientas que utilicen muchas personas que hacen trabajos similares pero no necesariamente iguales. El papel del analista de dominio<sup>5</sup> es descubrir y definir patrones de análisis, clases de análisis e información relacionada que pueda ser utilizada por mucha gente que trabaje en aplicaciones similares, pero que no son necesariamente las mismas.

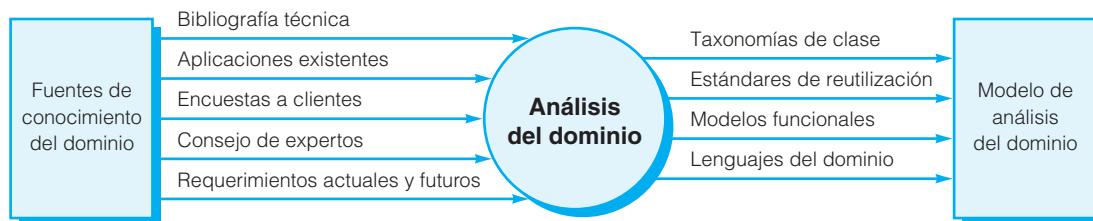
La figura 6.2 [Ara89] ilustra entradas y salidas clave para el proceso de análisis del dominio. Las fuentes de conocimiento del dominio se mapean con el fin de identificar los objetos que pueden reutilizarse a través del dominio.

**WebRef** En la dirección [www.iturls.com/English/SoftwareEngineering/SE\\_mod5.asp](http://www.iturls.com/English/SoftwareEngineering/SE_mod5.asp), existen muchos recursos útiles para el análisis del dominio.

**punto clave** El análisis del dominio no busca en una aplicación específica, sino en el dominio en el que reside la aplicación. El objetivo es identificar elementos comunes para la solución de problemas, que sean útiles en todas las aplicaciones dentro del dominio.

**cita:** “... el análisis es frustrante, está lleno de relaciones interpersonales complejas, indefinidas y difíciles. En una palabra, es fascinante. Una vez atrapado, los antiguos y fáciles placeres de la construcción de sistemas nunca más volverán a satisfacerte.”

**Tom DeMarco**



**FIGURA 6.2** Entradas y salidas para el análisis del dominio.

- 4 Un punto de vista complementario del análisis del dominio “involucra el modelado de éste, de manera que los ingenieros del software y otros participantes aprendan más al respecto [...] no todas las clases de dominio necesariamente dan como resultado el desarrollo de clases reutilizables [...]” [Let03a].
- 5 No suponga que el ingeniero de software no necesita entender el dominio de la aplicación tan sólo porque hay un analista del dominio trabajando. Todo miembro del equipo del software debe entender algo del dominio en el que se va a colocar el software.



## Análisis del dominio

**La escena:** Oficina de Doug Miller, después de una reunión con personal de mercadotecnia.

**Participantes:** Doug Miller, gerente de ingeniería de software, y Vinod Raman, miembro del equipo de ingeniería de software.

**La conversación:**

**Doug:** Te necesito para un proyecto especial, Vinod. Voy a retirarte de las reuniones para recabar los requerimientos.

**Vinod (con el ceño fruncido):** Muy mal. Ese formato en verdad funciona... Estaba sacando algo de ahí. ¿Qué pasa?

**Doug:** Jamie y Ed te cubrirán. De cualquier manera, el departamento de mercadotecnia insiste en que en la primera entrega de *CasaSegura* dispongamos de la capacidad de acceso por internet junto con la función de seguridad para el hogar. Estamos bajo fuego en esto... sin tiempo ni personal suficiente, así que tenemos que resolver ambos problemas a la vez: la interfaz de PC y la interfaz de web.

**Vinod (confundido):** No sabía que el plan era entregar... ni siquiera hemos terminado de recabar los requerimientos.

**Doug (con una sonrisa tenue):** Lo sé, pero los plazos son tan breves que decidí comenzar ya la estrategia con mercadotecnia... de cualquier modo, revisaremos cualquier plan tentativo una vez que tengamos la información de todas las juntas que se efectuarán para recabar los requerimientos.

**Vinod:** Está bien, ¿entonces? ¿Quéquieres que haga?

**Doug:** ¿Sabes qué es el "análisis del dominio"?

**Vinod:** Algo sé. Buscas patrones similares en aplicaciones que hagan lo mismo que la que estés elaborando. Entonces, si es posible, calcas los patrones y los reutilizas en tu trabajo.

**Doug:** No estoy seguro de que la palabra sea *calcar*, pero básicamente tienes razón. Lo que me gustaría que hicieras es que comenzaras a buscar interfaces de usuario ya existentes para sistemas que controlen algo como *CasaSegura*. Quiero que propongas un conjunto de patrones y clases de análisis que sean comunes tanto a la interfaz basada en PC que estará en el hogar como a la basada en un navegador al que se accederá por internet.

**Vinod:** Ahorraríamos tiempo si las hicieramos iguales... ¿por qué no las hacemos así?

**Doug:** Ah... es grato tener gente que piense como lo haces tú. Ése es el meollo del asunto: ahorraremos tiempo y esfuerzo si las dos interfaces son casi idénticas; las implementamos con el mismo código y acabamos con la insistencia de mercadotecnia.

**Vinod:** ¿Entonces, quéquieres?, ¿clases, patrones de análisis, patrones de diseño?

**Doug:** Todo eso. Nada formal en este momento. Sólo quiero que comencemos despacio con nuestros trabajos de análisis interno y de diseño.

**Vinod:** Iré a nuestra biblioteca de clases y veré qué tenemos. También usaré un formato de patrones que vi en un libro que leí hace unos meses.

**Doug:** Bien. Manos a la obra.

### > 6.1.4 Enfoques del modelado de requerimientos

Un enfoque del modelado de requerimientos, llamado *análisis estructurado*, considera que los datos y los procesos que los transforman son entidades separadas. Los objetos de datos se modelan de modo que se definen sus atributos y relaciones. Los procesos que manipulan a los objetos de datos

se modelan en forma que se muestre cómo transforman a los datos a medida que los objetos que se corresponden con ellos fluyen por el sistema.

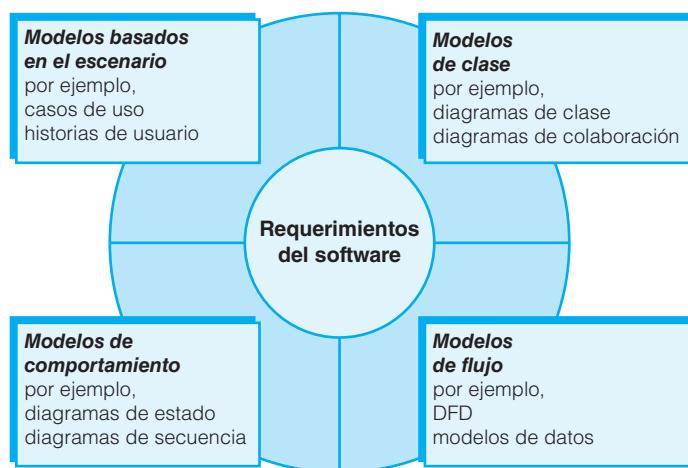
Un segundo enfoque del modelado del análisis, llamado *análisis orientado a objetos*, se centra en la definición de las clases y en la manera en la que colaboran uno con el otro para cumplir los requerimientos. El UML y el proceso unificado (véase el capítulo 2) están orientados a objetos, sobre todo.

Aunque el modelo de requerimientos propuesto en este libro combina características de ambos enfoques, los equipos de software escogen con frecuencia uno y excluyen todas las representaciones del otro. La pregunta no es cuál es mejor, sino qué combinación de representaciones proporcionará a los participantes el mejor modelo de requerimientos del software y el puente más eficaz para el diseño del mismo.

Cada elemento del modelo de requerimientos (véase la figura 6.3) presenta el problema desde diferentes puntos de vista. Los elementos basados en el escenario ilustran cómo interactúa el usuario con el sistema y la secuencia específica de actividades que ocurren cuando se utiliza el software. Los elementos basados en la clase modelan los objetos que el sistema manipulará, las operaciones que se aplicarán a ellos para realizar dicha manipulación, las relaciones (algunas jerárquicas) entre los objetos y las colaboraciones que ocurrirán entre las clases que se definen. Los elementos del comportamiento ilustran la forma en la que los eventos externos cambian el estado del sistema o las clases que residen dentro de éste. Por último, los elementos orientados al flujo representan al sistema como una transformación de la información e ilustran la forma en la que se transforman los objetos de datos cuando fluyen a través de las distintas funciones del sistema.

**cita:** “¿Por qué construimos modelos? ¿Por qué no construir sólo el sistema? La respuesta es que los construimos para que resalten o enfaticen ciertas características críticas de un sistema, al tiempo que ignoran otros aspectos del mismo.”

**Ed Yourdon**



¿Cuáles son los diferentes puntos de vista que se usan para describir el modelo de requerimientos?

**FIGURA 6.3** Elementos del modelo de análisis.

El modelado del análisis lleva a la obtención de cada uno de estos elementos de modelado. Sin embargo, el contenido específico de cada elemento (por ejemplo, los diagramas que se emplean para construir el elemento y el modelo) tal vez difiera de un proyecto a otro. Como se ha dicho varias veces en este libro, el equipo del software debe trabajar para mantenerlo sencillo. Sólo deben usarse elementos de modelado que agreguen valor al modelo.

## 6.2 Modelado basado en escenarios

Aunque el éxito de un sistema o producto basado en computadora se mide de muchas maneras, la satisfacción del usuario ocupa el primer lugar de la lista. Si se entiende cómo desean interactuar los usuarios finales (y otros actores) con un sistema, el equipo del software estará mejor preparado para caracterizar adecuadamente los requerimientos y hacer análisis significativos y modelos del diseño.

**cita:** “[Los casos de uso] simplemente son una ayuda para definir lo que existe fuera del sistema (actores) y lo que debe realizar el sistema (casos de uso).”

Ivar Jacobson

**consejo** En ciertas situaciones, los casos de uso se convierten en el mecanismo dominante de la ingeniería de requerimientos. Sin embargo, esto no significa que deban descartarse otros métodos de modelado cuando resulten apropiados.

Entonces, el modelado de los requerimientos con UML<sup>6</sup> comienza con la creación de escenarios en forma de casos de uso, diagramas de actividades y diagramas tipo carril de natación.

### 6.2.1 Creación de un caso preliminar de uso

Alistair Cockburn caracteriza un caso de uso como un “contrato para el comportamiento” [Coc01b]. Como se dijo en el capítulo 5, el “contrato” define la forma en la que un actor<sup>7</sup> utiliza un sistema basado en computadora para alcanzar algún objetivo. En esencia, un caso de uso capta las interacciones que ocurren entre los productores y consumidores de la información y el sistema en sí. En esta sección se estudiará la forma en la que se desarrollan los casos de uso como parte de los requerimientos de la actividad de modelado.<sup>8</sup>

En el capítulo 5 se dijo que un caso de uso describe en lenguaje claro un escenario específico desde el punto de vista de un actor definido. Pero, ¿cómo se sabe sobre qué escribir, cuánto escribir sobre ello, cuán detallada hacer la descripción y cómo organizarla? Son preguntas que deben responderse si los casos de uso han de tener algún valor como herramienta para modelar los requerimientos.

**:Sobre qué escribir?** Las dos primeras tareas de la ingeniería de requerimientos —concepción e indagación— dan la información que se necesita para comenzar a escribir casos de uso. Las reuniones para recabar los requerimientos, el DEC, y otros mecanismos para obtenerlos se utilizan para identificar a los participantes, definir el alcance del problema, especificar los objetivos operativos generales, establecer prioridades, delinejar todos los requerimientos funcionales conocidos y describir las cosas (objetos) que serán manipuladas por el sistema.

Para comenzar a desarrollar un conjunto de casos de uso, se enlistan las funciones o actividades realizadas por un actor específico. Éstas se obtienen de una lista de las funciones requeridas del sistema, por medio de conversaciones con los participantes o con la evaluación de los diagramas de actividades (véase la sección 6.3.1) desarrollados como parte del modelado de los requerimientos.



### Desarrollo de otro escenario preliminar de uso

**La escena:** Sala de juntas, durante la segunda reunión para recabar los requerimientos.

**Participantes:** Jamie Lazar, miembro del equipo del software; Ed Robbins, integrante del equipo del software; Doug Miller, gerente de ingeniería de software; tres miembros de mercadotecnia; un representante de ingeniería del producto, y un facilitador.

#### La conversación:

**Facilitador:** Es hora de que hablemos sobre la función de vigilancia de CasaSegura. Vamos a desarrollar un escenario de usuario que accede a la función de vigilancia.

**Jamie:** ¿Quién juega el papel del actor aquí?

**Facilitador:** Creo que Meredith (persona de mercadotecnia) ha estado trabajando en dicha funcionalidad. ¿Por qué no adoptas tú ese papel?

**Meredith:** Quieres que lo hagamos de la misma forma que la vez pasada, ¿verdad?

**Facilitador:** Sí... en cierto modo.

**Meredith:** Bueno, es obvio que la razón de la vigilancia es permitir que el propietario de la casa la revise cuando se encuentre fuera, así como poder grabar y reproducir el video que se grabe... esa clase de cosas.

6 En todo el libro se usará UML como notación para elaborar modelos. En el apéndice 1 se ofrece un método breve de enseñanza para aquellos lectores que no estén familiarizados con lo más básico de dicha notación.

7 Un actor no es una persona específica sino el rol que desempeña ésta (o un dispositivo) en un contexto específico. Un actor “llama al sistema para que entregue uno de sus servicios” [Coc01b].

8 Los casos de uso son una parte del modelado del análisis de importancia especial para las interfaces. El análisis de la interfaz se estudia en detalle en el capítulo 11.

**Ed:** ¿Usaremos compresión para guardar el video?

**Facilitador:** Buena pregunta, Ed, pero por ahora pospondremos los aspectos de la implementación. ¿Meredith?

**Meredith:** Bien, básicamente hay dos partes en la función de vigilancia... la primera configura el sistema, incluso un plano de la planta —tiene que haber herramientas que ayuden al propietario a hacer esto—, y la segunda parte es la función real de vigilancia. Como el plano es parte de la actividad de configuración, me centraré en la función de vigilancia.

**Facilitador (sonríe):** Me quitaste las palabras de la boca.

**Meredith:** Mmm... quiero tener acceso a la función de vigilancia, ya sea por PC o por internet. Tengo la sensación de que el acceso por internet se usaría con más frecuencia. De cualquier manera, quisiera poder mostrar vistas de la cámara en una PC y controlar el ángulo y acercamiento de una cámara en particular. Especificaría la cámara seleccionándola en el plano de la casa. También quiero poder bloquear el acceso a una o más cámaras con una clave determinada. Además, desearía tener la opción de ver pequeñas ventanas con vistas de todas las cámaras y luego escoger una que desee agrandar.

**Jamie:** Ésas se llaman vistas reducidas.

**Meredith:** Bien, entonces quiero vistas reducidas de todas las cámaras. También quisiera que la interfaz de la función de vigilancia tuviera el mismo aspecto y sensación que todas las demás del sistema *CasaSegura*. Quiero que sea intuitiva, lo que significa que no tenga que leer un manual para usarla.

**Facilitador:** Buen trabajo. Ahora, veamos esta función con un poco más de detalle...

La función (subsistema) de vigilancia de *CasaSegura* estudiada en el recuadro identifica las funciones siguientes (lista abreviada) que va a realizar el actor **propietario**:

- Seleccionar cámara para ver.
- Pedir vistas reducidas de todas las cámaras.
- Mostrar vistas de las cámaras en una ventana de PC.
- Controlar el ángulo y acercamiento de una cámara específica.
- Grabar la salida de cada cámara en forma selectiva.
- Reproducir la salida de una cámara.
- Acceder por internet a la vigilancia con cámaras.

A medida que avanzan las conversaciones con el participante (quien juega el papel de propietario), el equipo que recaba los requerimientos desarrolla casos de uso para cada una de las funciones estudiadas. En general, los casos de uso se escriben primero en forma de narración informal. Si se requiere más formalidad, se reescribe el mismo caso con el empleo de un formato estructurado, similar al propuesto en el capítulo y que se reproduce en un recuadro más adelante, en esta sección.

Para ilustrar esto, considere la función *acceder a la vigilancia con cámaras por internet-mostrar vistas de cámaras (AVC-MVC)*. El participante que tenga el papel del actor llamado **propietario** escribiría una narración como la siguiente:

**Caso de uso: acceder a la vigilancia con cámaras por internet, mostrar vistas de cámaras (AVC-MVC)**

**Actor: propietario**

Si estoy en una localidad alejada, puedo usar cualquier PC con un software de navegación apropiado para entrar al sitio web de *Productos CasaSegura*. Introduzco mi identificación de usuario y dos niveles de claves; una vez validadas, tengo acceso a toda la funcionalidad de mi sistema instalado. Para acceder a la vista de una cámara específica, selecciono “vigilancia” de los botones mostrados para las funciones principales. Luego selecciono “escoger una cámara” y aparece el plano de la casa. Despues elijo la cámara que me interesa. Alternativamente,

puedo ver la vista de todas las cámaras simultáneamente si selecciono “todas las cámaras”. Una vez que escojo una, selecciono “vista” y en la ventana que cubre la cámara aparece una vista con velocidad de un cuadro por segundo. Si quiero cambiar entre las cámaras, selecciono “escoger una cámara” y desaparece la vista original y de nuevo se muestra el plano de la casa. Después, selecciono la cámara que me interesa. Aparece una nueva ventana de vistas.

Una variación de la narrativa del caso de uso presenta la interacción como una secuencia ordenada de acciones del usuario. Cada acción está representada como enunciado declarativo. Al visitar la función **ACS-DCV**, se escribiría lo siguiente:

**Caso de uso: acceder a la vigilancia con cámaras por internet, mostrar vistas de cámaras (AVC-MVC)**

**Actor: propietario**

**cita:** “Los casos de uso se emplean en muchos procesos [de software]. Nuestro favorito es el que es iterativo y guiado por el riesgo.”

**Gerl Schneider y Jason Winters**

1. El propietario accede al sitio web *Productos CasaSegura*.
2. El propietario introduce su identificación de usuario.
3. El propietario escribe dos claves (cada una de al menos ocho caracteres de longitud).
4. El sistema muestra los botones de todas las funciones principales.
5. El propietario selecciona “vigilancia” de los botones de las funciones principales.
6. El propietario elige “seleccionar una cámara”.
7. El sistema presenta el plano de la casa.
8. El propietario escoge el ícono de una cámara en el plano de la casa.
9. El propietario selecciona el botón “vista”.
10. El sistema presenta la ventana de vista identificada con la elección de la cámara.
11. El sistema muestra un video dentro de la ventana a velocidad de un cuadro por segundo.

Es importante observar que esta presentación en secuencia no considera interacciones alternativas (la narración fluye con más libertad y representa varias alternativas). Los casos de este tipo en ocasiones se denominan *escenarios primarios* [Sch98a].

### > 6.2.2 Mejora de un caso de uso preliminar

Para entender por completo la función que describe un caso de uso, es esencial describir interacciones alternativas. Despues se evalúa cada paso en el escenario primario, planteando las preguntas siguientes [Sch98a]:

?

Cuando desarrollo un caso de uso, ¿cómo examino los cursos alternativos de acción?

- *¿El actor puede emprender otra acción en este punto?*
- *¿Es posible que el actor encuentre alguna condición de error en este punto?* Si así fuera, ¿cuál podría ser?
- *En este punto, ¿es posible que el actor encuentre otro comportamiento (por ejemplo, alguno que sea invocado por cierto evento fuera del control del actor)?* En ese caso, ¿cuál sería?

Las respuestas a estas preguntas dan como resultado la creación de un conjunto de *escenarios secundarios* que forman parte del caso de uso original, pero que representan comportamientos alternativos. Por ejemplo, considere los pasos 6 y 7 del escenario primario ya descrito:

6. El propietario elige “seleccionar una cámara”.
7. El sistema presenta el plano de la casa.

*¿El actor puede emprender otra acción en este punto?* La respuesta es “sí”. Al analizar la narración de flujo libre, el actor puede escoger mirar vistas de todas las cámaras simultáneamente. Entonces, un escenario secundario sería “observar vistas instantáneas de todas las cámaras”.

*¿Es posible que el actor encuentre alguna condición de error en este punto?* Cualquier número de condiciones de error puede ocurrir cuando opera un sistema basado en computadora. En este con-

texto, sólo se consideran las condiciones que sean probables como resultado directo de la acción descrita en los pasos 6 o 7. De nuevo, la respuesta es “sí”. Tal vez nunca se haya configurado un plano con íconos de cámara. Entonces, elegir “seleccionar una cámara” da como resultado una condición de error: “No hay plano configurado para esta casa.”<sup>9</sup> Esta condición de error se convierte en un escenario secundario.

*En este punto, ¿es posible que el actor encuentre otro comportamiento (por ejemplo, alguno que sea invocado por cierto evento fuera del control del actor)?* Otra vez, la respuesta es “sí”. A medida que ocurran los pasos 6 y 7, el sistema puede hallar una condición de alarma. Esto dará como resultado que el sistema desplegará una notificación especial de alarma (tipo, ubicación, acción del sistema) y proporcionará al actor varias opciones relevantes según la naturaleza de la alarma. Como este escenario secundario puede ocurrir en cualquier momento para prácticamente todas las interacciones, no se vuelve parte del caso de uso **AVC-MVC**. En vez de ello, se desarrollará un caso de uso diferente —**Condición de alarma encontrada**— al que se hará referencia desde otros casos según se requiera.

Cada una de las situaciones descritas en los párrafos precedentes se caracteriza como una excepción al caso de uso. Una *excepción* describe una situación (ya sea condición de falla o alternativa elegida por el actor) que ocasiona que el sistema presente un comportamiento algo distinto.

Cockburn [Coc01b] recomienda el uso de una sesión de “lluvia de ideas” para obtener un conjunto razonablemente complejo de excepciones para cada caso de uso. Además de las tres preguntas generales ya sugeridas en esta sección, también deben explorarse los siguientes aspectos:

- *¿Existen casos en los que ocurra alguna “función de validación” durante este caso de uso?* Esto implica que la función de validación es invocada y podría ocurrir una potencial condición de error.
- *¿Hay casos en los que una función (o actor) de soporte falle en responder de manera apropiada?* Por ejemplo, una acción de usuario espera una respuesta pero la función que ha de responder se cae.
- *¿El mal desempeño del sistema da como resultado acciones inesperadas o impropias?* Por ejemplo, una interfaz con base en web responde con demasiada lentitud, lo que da como resultado que un usuario haga selecciones múltiples en un botón de procesamiento. Estas selecciones se forman de modo equivocado y, en última instancia, generan un error.

La lista de extensiones desarrollada como consecuencia de preguntar y responder estas preguntas debe “racionalizarse” [Coc01b] con el uso de los siguientes criterios: una excepción debe describirse dentro del caso de uso si el software la puede detectar y debe manejarla una vez detectada. En ciertos casos, una excepción precipitará el desarrollo de otro caso de uso (el de manejar la condición descrita).

### > 6.2.3 Escritura de un caso de uso formal

En ocasiones, para modelar los requerimientos es suficiente con los casos de uso informales presentados en la sección 6.2.1. Sin embargo, cuando un caso de uso involucra una actividad crítica o cuando describe un conjunto complejo de etapas con un número significativo de excepciones, es deseable un enfoque más formal.

El caso de uso **AVC-MVC** mostrado en el recuadro de la página 136 sigue el guión común para los casos de uso formales. El *objetivo en contexto* identifica el alcance general del caso de uso. La *precondición* describe lo que se sabe que es verdadero antes de que inicie el caso de uso. El *disparador* (o *trigger*) identifica el evento o condición que “hace que comience el caso de uso” [Coc01b]. El *escenario* enumera las acciones específicas que requiere el actor, y las respuestas apropiadas del sistema. Las *excepciones* identifican las situaciones detectadas cuando se mejora el caso de uso preliminar (véase la sección 6.2.2). Pueden incluirse o no encabezados adicionales y se explican por sí mismos en forma razonable.

---

<sup>9</sup> En este caso, otro actor, **administrador del sistema**, tendría que configurar el plano de la casa, instalar e inicializar todas las cámaras (por ejemplo, asignar una identificación a los equipos) y probar cada una para garantizar que se encuentren accesibles por el sistema y a través del plano de la casa.



## Formato de caso de uso para vigilancia

Caso de uso: Acceder a la vigilancia con cámaras por internet, mostrar vistas de cámaras (AVC-MVC).

**Iteración:**

2, última modificación: 14 de enero por V. Raman.

**Actor principal:**

Propietario.

**Objetivo en contexto:**

Ver la salida de las cámaras colocadas en la casa desde cualquier ubicación remota por medio de internet.

**Precondiciones:**

El sistema debe estar configurado por completo; deben obtenerse las identificaciones y claves de usuario apropiadas.

**Disparador:**

El propietario decide ver dentro de la casa mientras está fuera.

**Escenario:**

1. El propietario se registra en el sitio web *Productos CasaSegura*.
2. El propietario introduce su identificación de usuario.
3. El propietario proporciona dos claves (cada una con longitud de al menos ocho caracteres).
4. El sistema despliega todos los botones de las funciones principales.
5. El propietario selecciona "vigilancia" entre los botones de funciones principales.
6. El propietario escoge "seleccionar una cámara".
7. El sistema muestra el plano de la casa.
8. El propietario selecciona un ícono de cámara en el plano de la casa.
9. El propietario pulsa el botón "vista".
10. El sistema muestra la ventana de la vista de la cámara identificada.
11. El sistema presenta una salida de video dentro de la ventana de vistas, con una velocidad de un cuadro por segundo.

**Excepciones:**

1. La identificación o las claves son incorrectas o no se reconocen (véase el caso de uso **Validar identificación y claves**).
2. La función de vigilancia no está configurada para este sistema (el sistema muestra el mensaje de error apropiado; véase el caso de uso **Configurar la función de vigilancia**).
3. El propietario selecciona "Mirar vistas reducidas de todas las cámaras" (véase el caso de uso **Mirar vistas reducidas de todas las cámaras**).
4. No se dispone o no se ha configurado el plano de la casa (se muestra el mensaje de error apropiado y véase el caso de uso **Configurar plano de la casa**).
5. Se encuentra una condición de alarma (véase el caso de uso **Condición de alarma encontrada**).

**Prioridad:**

Moderada, por implementarse después de las funciones básicas.

**Cuándo estará disponible:** En el tercer incremento.

**Frecuencia de uso:** Frecuencia moderada.

**Canal al actor:** A través de un navegador con base en PC y conexión a internet.

**Actores secundarios:**

Administrador del sistema, cámaras.

### Canales a los actores secundarios:

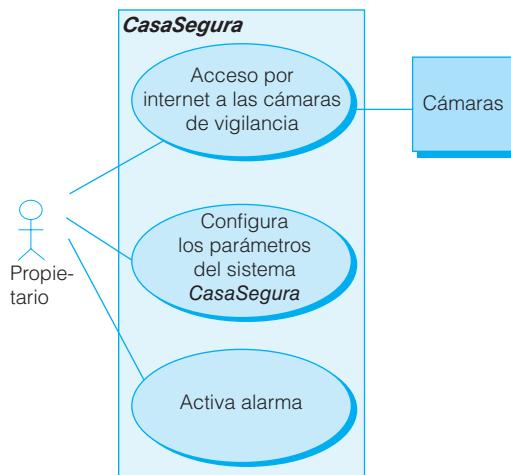
1. Administrador del sistema: sistema basado en PC.
2. Cámaras: conectividad inalámbrica.

### Asuntos pendientes:

1. ¿Qué mecanismos protegen el uso no autorizado de esta capacidad por parte de los empleados de *Productos CasaSegura*?
2. Es suficiente la seguridad? El acceso ilegal a esta característica representaría una invasión grave de la privacidad.
3. ¿Será aceptable la respuesta del sistema por internet dado el ancho de banda que requieren las vistas de las cámaras?
4. ¿Desarrollaremos una capacidad que provea el video a una velocidad más alta en cuadros por segundo cuando se disponga de conexiones con un ancho de banda mayor?

En muchos casos, no hay necesidad de crear una representación gráfica de un escenario de uso. Sin embargo, la representación con diagramas facilita la comprensión, en particular cuando el escenario es complejo. Como ya se dijo en este libro, UML cuenta con la capacidad de hacer diagramas de casos de uso. La figura 6.4 ilustra un diagrama de caso de uso preliminar para el producto *CasaSegura*. Cada caso de uso está representado por un óvalo. En esta sección sólo se estudia el caso de uso **AVC-MVC**.

**WebRef** ¿Cuándo se ha terminado de escribir casos de uso? Para un análisis benéfico de esto, consulte la dirección [otips.org/use-cases-done.html](http://otips.org/use-cases-done.html)



**FIGURA 6.4** Diagrama de caso de uso preliminar para el sistema *CasaSegura*.

Toda notación de modelado tiene sus limitaciones, y la del caso de uso no es la excepción. Como cualquier otra forma de descripción escrita, un caso de uso es tan bueno como lo sea(n) su(s) autor(es). Si la descripción es poco clara, el caso de uso será confuso o ambiguo. Un caso de uso se centra en los requerimientos funcionales y de comportamiento, y por lo general es inapropiado para requerimientos disfuncionales. Para situaciones en las que el modelo de requerimientos deba tener detalle y precisión significativos (por ejemplo, sistemas críticos de seguridad), tal vez no sea suficiente un caso de uso.

Sin embargo, el modelado basado en escenarios es apropiado para la gran mayoría de todas las situaciones que encontrará un ingeniero de software. Si se desarrolla bien, el caso de uso proporciona un beneficio sustancial como herramienta de modelado.

## 6.3 Modelos UML que proporcionan el caso de uso

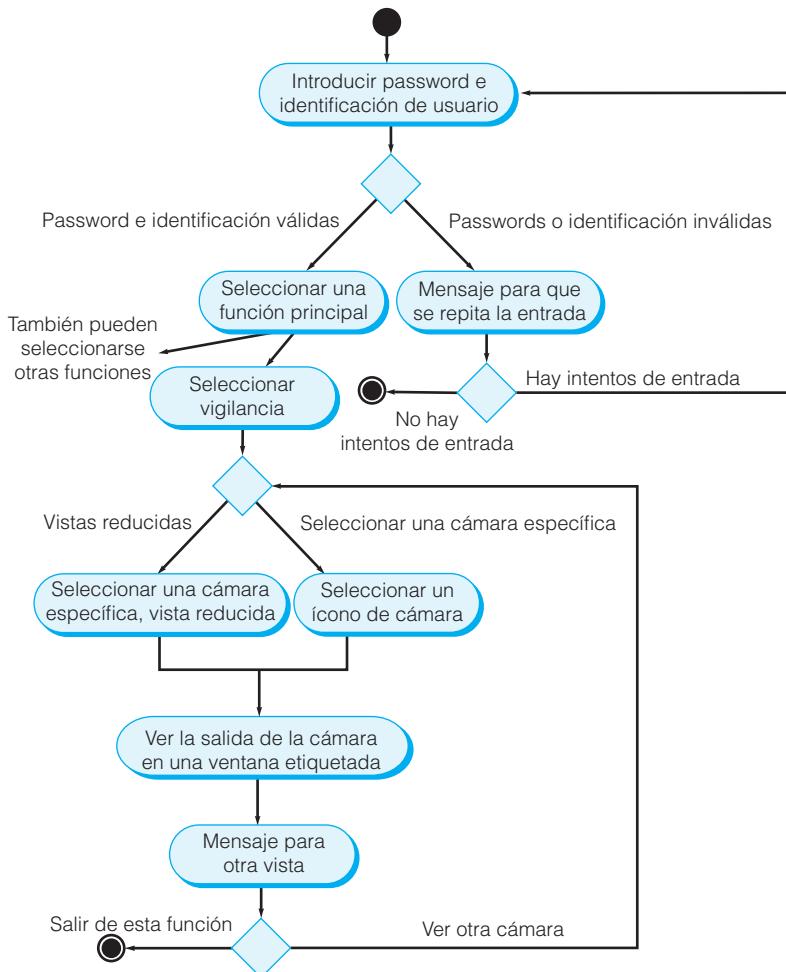
Hay muchas situaciones de modelado de requerimientos en las que un modelo basado en texto — incluso uno tan sencillo como un caso de uso— tal vez no brinde información en forma clara y concisa. En tales casos, es posible elegir de entre una amplia variedad de modelos UML gráficos.

### > 6.3.1 Desarrollo de un diagrama de actividades

**punto clave** Un diagrama de actividades UML representa las acciones y decisiones que ocurren cuando se realiza cierta función.

El diagrama de actividad UML enriquece el caso de uso al proporcionar una representación gráfica del flujo de interacción dentro de un escenario específico. Un diagrama de actividades es similar a uno de flujo, y utiliza rectángulos redondeados para denotar una función específica del sistema, flechas para representar flujo a través de éste, rombos de decisión para ilustrar una ramificación de las decisiones (cada flecha que salga del rombo se etiqueta) y líneas continuas para indicar que están ocurriendo actividades en paralelo. En la figura 6.5 se presenta un diagrama de actividades para el caso de uso **AVC-MVC**. Debe observarse que el diagrama de actividades agrega detalles adicionales que no se mencionan directamente (pero que están implícitos) en el caso de uso.

Por ejemplo, un usuario quizás sólo haga algunos intentos de introducir su **identificación** y **password**. Esto se representa por el rombo de decisión debajo de “Mensaje para que se repita la entrada”.



**FIGURA 6.5** Diagrama de actividades para la función Acceder a la vigilancia con cámaras por internet, mostrar vistas de cámaras.

### > 6.3.2 Diagramas de canal (swimlane)

El *diagrama de canal* de UML es una variación útil del diagrama de actividades y permite representar el flujo de actividades descritas por el caso de uso; al mismo tiempo, indica qué actor (si hubiera muchos involucrados en un caso específico de uso) o clase de análisis (se estudia más adelante, en este capítulo) es responsable de la acción descrita por un rectángulo de actividad. Las responsabilidades se representan con segmentos paralelos que dividen el diagrama en forma vertical, como los canales o carriles de una alberca.

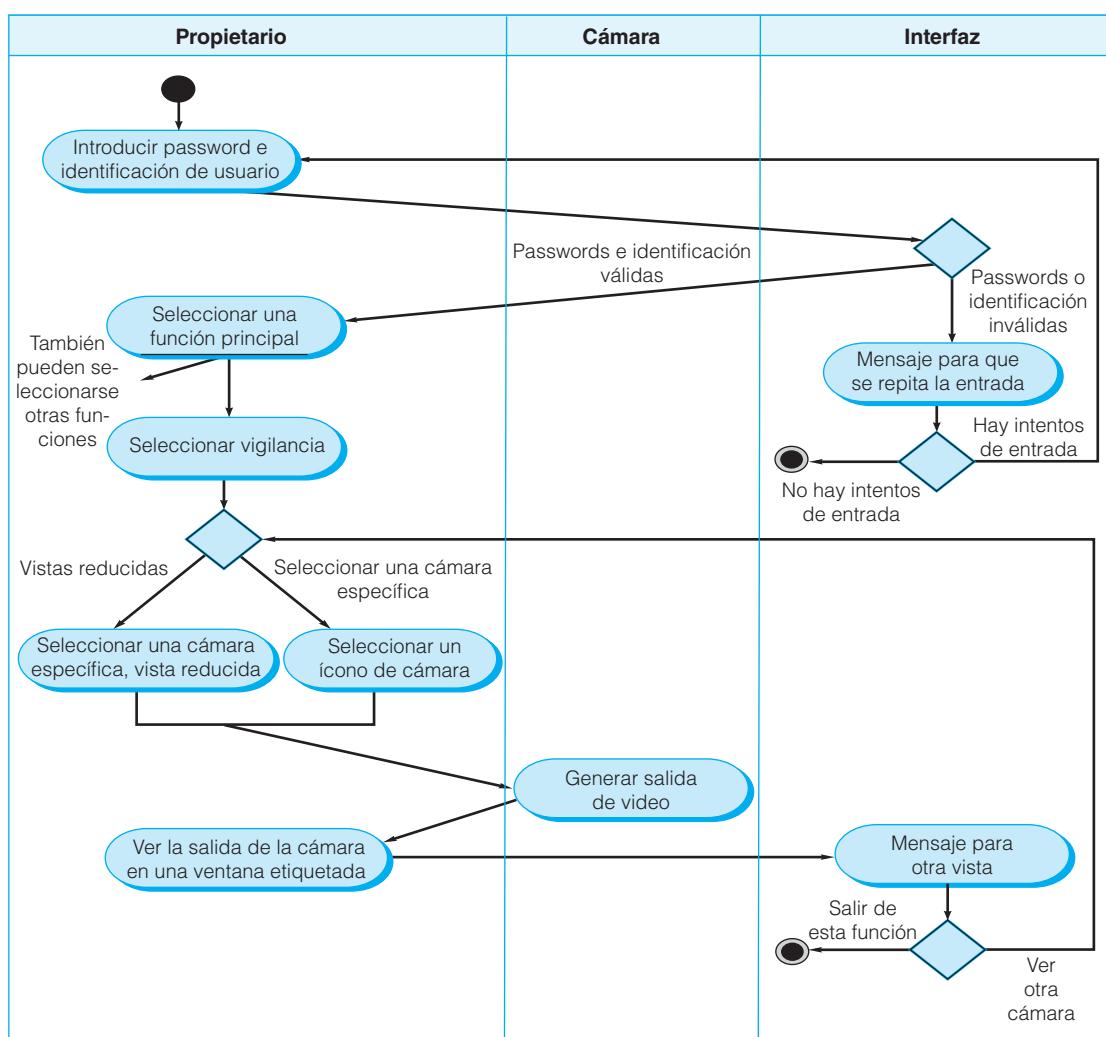
Son tres las clases de análisis: **Propietario**, **Cámara** e **Interfaz**, que tienen responsabilidad directa o indirecta en el contexto del diagrama de actividades representado en la figura 6.5. En relación con la figura 6.6, el diagrama de actividades se reacomodó para que las actividades asociadas con una clase de análisis particular queden dentro del canal de dicha clase. Por ejemplo, la clase **Interfaz** representa la interfaz de usuario como la ve el propietario. El diagrama de actividades tiene dos mensajes que son responsabilidad de la interfaz: “mensaje para que se repita la entrada” y “mensaje para otra vista”. Estos mensajes y las decisiones asociadas con ellos caen dentro del canal **Interfaz**. Sin embargo, las flechas van de ese canal de regreso al de **Propietario**, donde ocurren las acciones de éste.

Los casos de uso, junto con los diagramas de actividades y de canal, están orientados al procedimiento. Representan la manera en la que los distintos actores invocan funciones específicas (u otros

**punto clave** Un diagrama de canal (swimlane) representa el flujo de acciones y decisiones e indica qué actores efectúan cada una de ellas.

**cita:** “Un buen modelo guía el pensamiento; uno malo lo desvía.”

Brian Marick



**FIGURA 6.6** Diagrama de canal para la función Acceder a la vigilancia con cámaras por internet, mostrar vistas de cámaras.

pasos del procedimiento) para satisfacer los requerimientos del sistema. Pero una vista del procedimiento de los requerimientos representa una sola dimensión del sistema. En la sección 6.4 se estudia el espacio de información y la forma en la que se representan los datos de requerimientos.

## 6.4 Conceptos de modelado de datos

**WebRef** En la dirección [www.datamodel.org](http://www.datamodel.org), hay información útil sobre el modelado de datos.

Si los requerimientos del software incluyen la necesidad de crear, ampliar o hacer interfaz con una base de datos, o si deben construirse y manipularse estructuras de datos complejas, el equipo del software tal vez elija crear un *modelo de datos* como parte del modelado general de los requerimientos. Un ingeniero o analista de software define todos los objetos de datos que se procesan dentro del sistema, la relación entre ellos y otro tipo de información que sea pertinente para las relaciones. El *diagrama entidad-relación* (DER) aborda dichos aspectos y representa todos los datos que se introducen, almacenan, transforman y generan dentro de una aplicación.

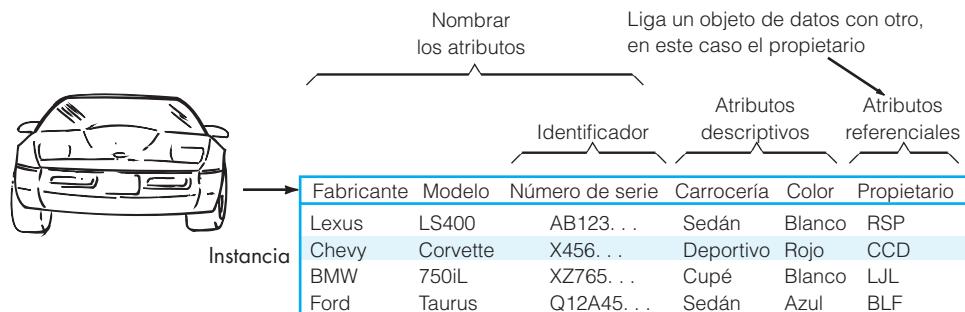
### > 6.4.1 Objetos de datos

¿Cómo se manifiesta un objeto de datos en el contexto de una aplicación?

Un *objeto de datos* es una representación de información compuesta que debe ser entendida por el software. *Información compuesta* quiere decir algo que tiene varias propiedades o atributos diferentes. Por tanto, el ancho (un solo valor) no sería un objeto de datos válido, pero las **dimensiones** (que incorporan altura, ancho y profundidad) sí podrían definirse como un objeto.

Un objeto de datos puede ser una entidad externa (por ejemplo, cualquier cosa que produzca o consuma información), una cosa (por ejemplo, un informe o pantalla), una ocurrencia (como una llamada telefónica) o evento (por ejemplo, una alarma), un rol (un vendedor), una unidad organizacional (por ejemplo, el departamento de contabilidad), un lugar (como una bodega) o estructura (como un archivo). Por ejemplo, una **persona** o un **auto** pueden considerarse como objetos de datos en tanto cada uno se define en términos de un conjunto de atributos. La descripción del objeto de datos incorpora a ésta todos sus atributos.

Un objeto de datos contiene sólo datos —dentro de él no hay referencia a las operaciones que se apliquen sobre los datos—.<sup>10</sup> Entonces, el objeto de datos puede representarse en forma de tabla, como la que se muestra en la figura 6.7. Los encabezados de la tabla reflejan atributos del objeto. En este caso, un auto se define en términos de *fabricante*, *modelo*, *número de serie*, *tipo de carrocería*, *color* y *propietario*. El cuerpo de la tabla representa instancias específicas del objeto de datos. Por ejemplo, un Chevy Corvette es una instancia del objeto de datos **auto**.



The diagram illustrates the concept of an object of data. On the left, there is a simple line drawing of a front-wheel-drive car. To its right, the word "Instancia" is written below it. An arrow points from the car icon to a table. Above the table, several labels with arrows point to specific parts: "Nombrar los atributos" points to the column headers; "Liga un objeto de datos con otro, en este caso el propietario" points to the "Propietario" column; "Identificador" points to the "Número de serie" column; "Atributos descriptivos" points to the "Fabricante", "Modelo", "Carrocería", "Color", and "Propietario" columns; and "Atributos referenciales" points to the "Número de serie" column again, indicating it links to another object (the owner).

Fabricante	Modelo	Número de serie	Carrocería	Color	Propietario
Lexus	LS400	AB123...	Sedán	Blanco	RSP
Chevy	Corvette	X456...	Deportivo	Rojo	CCD
BMW	750iL	XZ765...	Cupé	Blanco	LJL
Ford	Taurus	Q12A45...	Sedán	Azul	BLF

**FIGURA 6.7** Representación tabular de objetos de datos.

<sup>10</sup> Esta distinción separa al objeto de datos de la clase u objeto definidos como parte del enfoque orientado a objetos (véase el apéndice 2).

## > 6.4.2 Atributos de los datos

Los *atributos de los datos* definen las propiedades de un objeto de datos y tienen una de tres diferentes características. Se usan para 1) nombrar una instancia del objeto de datos, 2) describir la instancia o 3) hacer referencia a otra instancia en otra tabla. Además, debe definirse como identificador uno o más de los atributos —es decir, el atributo identificador se convierte en una “llave” cuando se desea encontrar una instancia del objeto de datos—. En ciertos casos, los valores para el (los) identificador(es) son únicos, aunque esto no es una exigencia. En relación con el objeto de datos **auto**, un identificador razonable sería el *número de serie*.

El conjunto de atributos que es apropiado para un objeto de datos determinado se define entendiendo el contexto del problema. Los atributos para **auto** podrían servir bien para una aplicación que usara un departamento de vehículos motorizados, pero serían inútiles para una compañía automotriz que necesitara hacer software de control de manufactura. En este último caso, los atributos para **auto** quizás también incluyan *número de serie, tipo de carrocería y color*, pero tendrían que agregarse muchos otros (por ejemplo, *código interior, tipo de tracción, indicador de paquete de recorte, tipo de transmisión*, etc.) para hacer de **auto** un objeto significativo en el contexto de control de manufactura.

**punto clave** Los atributos nombran a un objeto de datos, describen sus características y, en ciertos casos, hacen referencia a otro objeto.

**WebRef** Para aquellos que intentan hacer modelado de datos, es importante un concepto llamado “normalización”. En la dirección [www.datamodel.org](http://www.datamodel.org) se encuentra una introducción útil.



## Objetos de datos y clases orientadas a objetos: ¿son lo mismo?

Al analizar objetos de datos es común que surja una pregunta: ¿un objeto de datos es lo mismo que una clase orientada<sup>11</sup> a objetos? La respuesta es “no”.

Un objeto de datos define un aspecto de datos compuestos; es decir, incorpora un conjunto de características de datos individuales (atributos) y da al conjunto un nombre (el del objeto de datos).

Una clase orientada a objetos encierra atributos de datos, pero también incorpora las operaciones (métodos) que los manipulan y que están determinadas por dichos atributos. Además, la definición de clases implica una infraestructura amplia que es parte del enfoque de la ingeniería de software orientada a objetos. Las clases se comunican entre sí por medio de mensajes, se organizan en jerarquías y tienen características hereditarias para los objetos que son una instancia de una clase.

Información

## > 6.4.3 Relaciones

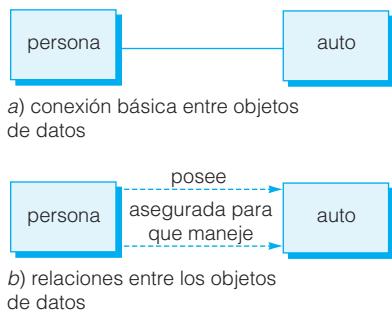
Los objetos de datos están conectados entre sí de diferentes maneras. Considere dos objetos de datos, **persona** y **auto**. Estos objetos se representan con la notación simple que se ilustra en la figura 6.8a). Se establece una conexión entre **persona** y **auto** porque ambos objetos están relacionados. Pero, ¿cuál es esa relación? Para determinarlo, debe entenderse el papel de las personas (propiedad, en este caso) y los autos dentro del contexto del software que se va a elaborar. Se establece un conjunto de parejas objeto/relación que definen las relaciones relevantes. Por ejemplo,

- Una persona *posee* un auto.
- Una persona *es asegurada para que maneje* un auto.

**punto clave** Las relaciones indican la manera en la que los objetos de datos se conectan entre sí.

Las relaciones *posee* y *es asegurada para que maneje* definen las conexiones relevantes entre **persona** y **auto**. La figura 6.8b) ilustra estas parejas objeto-relación. Las flechas en esa figura dan información importante sobre la dirección de la relación y es frecuente que reduzcan las ambigüedades o interpretaciones erróneas.

<sup>11</sup> Los lectores que no estén familiarizados con los conceptos y terminología de la orientación a objetos deben consultar el breve instructivo que se presenta en el apéndice 2.



**FIGURA 6.8** Relaciones entre objetos de datos.

## Información



### Diagramas entidad-relación

La pareja objeto-relación es la piedra angular del modelo de datos. Estas parejas se representan gráficamente con el uso del diagrama entidad-relación (DER).<sup>12</sup> El DER fue propuesto por primera vez por Peter Chen [Che77] para diseñar sistemas de bases de datos relacionales y ha sido ampliado por otras personas. Se identifica un conjunto de componentes primarios para el DER: objetos de datos, atributos, relaciones y distintos indicadores de tipo. El propósito principal del DER es representar objetos de datos y sus relaciones.

Ya se presentó la notación DER básica. Los objetos de datos se representan con un rectángulo etiquetado. Las relaciones se indican con una línea etiquetada que conecta objetos. En ciertas variantes del DER, la línea de conexión contiene un rombo con la leyenda de la relación. Las conexiones entre los objetos de datos y las relaciones se establecen con el empleo de varios símbolos especiales que indican cardinalidad y modalidad.<sup>13</sup> Si el lector está interesado en obtener más información sobre el modelado de datos y el diagrama entidad-relación, consulte [Hob06] o [Sim05].

## 6.5 Modelado basado en clases

El modelado basado en clases representa los objetos que manipulará el sistema, las operaciones (también llamadas *métodos* o *servicios*) que se aplicarán a los objetos para efectuar la manipulación, las relaciones (algunas de ellas jerárquicas) entre los objetos y las colaboraciones que tienen lugar entre las clases definidas. Los elementos de un modelo basado en clases incluyen las clases y los objetos, atributos, operaciones, modelos clase-responsabilidad-colaborador (CRC), diagramas de colaboración y paquetes. En las secciones siguientes se presenta una serie de lineamientos informales que ayudarán a su identificación y representación.

12 Aunque algunas aplicaciones de diseño de bases de datos aún emplean el DER, ahora se utiliza la notación UML (véase el apéndice 1) para el diseño de datos.

13 La *cardinalidad* de una pareja objeto-relación especifica “el número de ocurrencias de uno [objeto] que se relaciona con el número de ocurrencias de otro [objeto]” [Til93]. La *modalidad* de una relación es 0 si no hay necesidad explícita para que ocurra la relación o si ésta es opcional. La modalidad es 1 si una ocurrencia de la relación es obligatoria.



## Modelado de datos

**Objetivo:** Las herramientas de modelado de datos dan a un ingeniero de software la capacidad de representar objetos de datos, sus características y relaciones. Se usan sobre todo para aplicaciones de grandes bases de datos y otros proyectos de sistemas de información, y proveen medios automatizados para crear diagramas completos de entidad-relación, diccionarios de objetos de datos y modelos relacionados.

**Mecánica:** Las herramientas de esta categoría permiten que el usuario describa objetos de datos y sus relaciones. En ciertos casos, utilizan notación DER. En otros, modelan relaciones con el empleo de un mecanismo diferente. Es frecuente que las herramientas en esta categoría se usen como parte del diseño de una base de datos y que permitan la creación de su modelo con la generación de un esquema para sistemas comunes de administración de bases de datos comunes (DBMS).

### Herramientas representativas:<sup>14</sup>

AllFusion ERWin, desarrollada por Computer Associates ([www3.ca.com](http://www3.ca.com)), ayuda en el diseño de objetos de datos, estructura apropiada y elementos clave para las bases de datos.

ER/Studio, desarrollada por Embarcadero Software ([www.embarcadero.com](http://www.embarcadero.com)), da apoyo al modelado entidad-relación.

Oracle Designer, desarrollada por Oracle Systems ([www.oracle.com](http://www.oracle.com)), "modela procesos de negocios, entidades y relaciones de datos [que] se transforman en diseños para los que se generan aplicaciones y bases de datos completas".

Visible Analyst, desarrollada por Visible Systems ([www.visible.com](http://www.visible.com)), da apoyo a varias funciones de modelado del análisis, incluso modelado de datos.

# Herramientas de software

### > 6.5.1 Identificación de las clases de análisis

Al mirar una habitación, se observa un conjunto de objetos físicos que se identifican, clasifican y definen fácilmente (en términos de atributos y operaciones). Pero cuando se "ve" el espacio del problema de una aplicación de software, las clases (y objetos) son más difíciles de concebir.

Se comienza por identificar las clases mediante el análisis de los escenarios de uso desarrollados como parte del modelo de requerimientos y la ejecución de un "análisis gramatical" [Abb83] sobre los casos de uso desarrollados para el sistema que se va a construir. Las clases se determinan subrayando cada sustantivo o frase que las incluya para introducirlo en una tabla simple. Deben anotarse los sinónimos. Si la clase (sustantivo) se requiere para implementar una solución, entonces forma parte del espacio de solución; de otro modo, si sólo es necesaria para describir la solución, es parte del espacio del problema.

Pero, ¿qué debe buscarse una vez identificados todos los sustantivos? Las *clases de análisis* se manifiestan en uno de los modos siguientes:

- *Entidades externas* (por ejemplo, otros sistemas, dispositivos y personas) que producen o consumen la información que usará un sistema basado en computadora.
- *Cosas* (reportes, pantallas, cartas, señales, etc.) que forman parte del dominio de información para el problema.
- *Ocurrencias o eventos* (como una transferencia de propiedad o la ejecución de una serie de movimientos de un robot) que suceden dentro del contexto de la operación del sistema.
- *Roles* (gerente, ingeniero, vendedor, etc.) que desempeñan las personas que interactúan con el sistema.

**cita:** "El problema realmente difícil es descubrir en primer lugar cuáles son los objetos correctos [clases]."

**Carl Argila**

¿Cómo se manifiestan las clases en tantos elementos del espacio de solución?

<sup>14</sup> Las herramientas mencionadas aquí no son obligatorias sino una muestra de las que hay en esta categoría. En la mayoría de casos, los nombres de las herramientas son marcas registradas por sus respectivos desarrolladores.

- *Unidades organizacionales* (división, grupo, equipo, etc.) que son relevantes para una aplicación.
- *Lugares* (piso de manufactura o plataforma de carga) que establecen el contexto del problema y la función general del sistema.
- *Estructuras* (sensores, vehículos de cuatro ruedas, computadoras, etc.) que definen una clase de objetos o clases relacionadas de éstos.

Esta clasificación sólo es una de muchas propuestas en la bibliografía.<sup>15</sup> Por ejemplo, Budd [Bud96] sugiere una taxonomía de clases que incluye *productores* (fuentes) y *consumidores* (sumideros) de datos, *administradores de datos, vista, clases de observador y clases de auxiliares*.

También es importante darse cuenta de lo que no son las clases u objetos. En general, una clase nunca debe tener un “nombre de procedimiento imperativo” [Cas89]. Por ejemplo, si los desarrolladores del software de un sistema de imágenes médicas definieron un objeto con el nombre **InvertirImagen** o incluso **InversióndeImagen**, cometerían un error sutil. La **Imagen** obtenida del software podría ser, por supuesto, una clase (algo que es parte del dominio de la información). La inversión de la imagen es una operación que se aplica al objeto. Es probable que la inversión esté definida como una operación para el objeto **Imagen**, pero no lo estaría como clase separada con la connotación “inversión de imagen”. Como afirma Cashman [Cas89]: “el intento de la orientación a objetos es contener, pero mantener separados, los datos y las operaciones sobre ellos”.

Para ilustrar cómo podrían definirse las clases del análisis durante las primeras etapas del modelado, considere un análisis gramatical (los sustantivos están subrayados, los verbos aparecen en cursivas) de una narración de procesamiento<sup>16</sup> para la función de seguridad de *CasaSegura*.

La función de seguridad CasaSegura permite que el propietario configure el sistema de seguridad cuando se instala, vigila todos los sensores conectados al sistema de seguridad e interactúa con el propietario a través de internet, una PC o panel de control.

Durante la instalación, la PC de CasaSegura se utiliza para programar y configurar el sistema. Se asigna a cada sensor un número y tipo, se programa un password maestro para activar y desactivar el sistema y se introducen números telefónicos para marcar cuando ocurre un evento de sensor.

Cuando se reconoce un evento de sensor, el software invoca una alarma audible instalada en el sistema. Despues de un tiempo de retraso que especifica el propietario durante las actividades de configuración del sistema, el software marca un número telefónico de un servicio de monitoreo, proporciona información acerca de la ubicación y reporta la naturaleza del evento detectado. El número telefónico se vuelve a marcar cada 20 segundos hasta que se obtiene la conexión telefónica.

El propietario recibe información de seguridad a través de un panel de control, la PC o un navegador, lo que en conjunto se llama interfaz. La interfaz despliega mensajes de aviso e información del estado del sistema en el panel de control, la PC o la ventana del navegador. La interacción del propietario tiene la siguiente forma...

Con los sustantivos se proponen varias clases potenciales:

Clase potencial	Clasificación general
propietario	rol de entidad externa
sensor	entidad externa
panel de control	entidad externa
instalación	ocurrencia
sistema (alias sistema de seguridad)	cosa
número, tipo	no objetos, atributos de sensor

<sup>16</sup> Una narración de procesamiento es similar al caso de uso en su estilo, pero algo distinto en su propósito. La narración de procesamiento hace una descripción general de la función que se va a desarrollar. No es un escenario escrito desde el punto de vista de un actor. No obstante, es importante observar que el análisis gramatical también puede emplearse para todo caso de uso desarrollado como parte de la obtención de requerimientos (indagación).

password maestro	cosa
número telefónico	cosa
evento de sensor	ocurrencia
alarma audible	entidad externa
servicio de monitoreo externa	unidad organizacional o entidad

La lista continuará hasta que se hayan considerado todos los sustantivos en la narrativa de procesamiento. Observe que cada entrada en la lista se llama objeto potencial. El lector debe considerar cada una antes de tomar la decisión final.

Coad y Yourdon [Coa91] sugieren seis características de selección que deben usarse cuando se considere cada clase potencial para incluirla en el modelo de análisis:

1. *Información retenida.* La clase potencial será útil durante el análisis sólo si debe recordarse la información sobre ella para que el sistema pueda funcionar.
2. *Servicios necesarios.* La clase potencial debe tener un conjunto de operaciones identificables que cambien en cierta manera el valor de sus atributos.
3. *Atributos múltiples.* Durante el análisis de los requerimientos, la atención debe estar en la información “principal”; en realidad, una clase con un solo atributo puede ser útil durante el diseño, pero es probable que durante la actividad de análisis se represente mejor como un atributo de otra clase.
4. *Atributos comunes.* Para la clase potencial se define un conjunto de atributos y se aplican éstos a todas las instancias de la clase.
5. *Operaciones comunes.* Se define un conjunto de operaciones para la clase potencial y éstas se aplican a todas las instancias de la clase.
6. *Requerimientos esenciales.* Las entidades externas que aparezcan en el espacio del problema y que produzcan o consuman información esencial para la operación de cualquier solución para el sistema casi siempre se definirán como clases en el modelo de requerimientos.



¿Cómo determino si una clase potencial debe, en realidad, ser una clase de análisis?

Para que se considere una clase legítima para su inclusión en el modelo de requerimientos, un objeto potencial debe satisfacer todas (o casi todas) las características anteriores. La decisión de incluir clases potenciales en el modelo de análisis es algo subjetiva, y una evaluación posterior tal vez haga que un objeto se descarte o se incluya de nuevo. Sin embargo, el primer paso del modelado basado en clases es la definición de éstas, y deben tomarse las medidas respectivas (aun las subjetivas). Con esto en mente, se aplicarán las características de selección a la lista de clases potenciales de *CasaSegura*:

**cita:** “Las clases luchan, algunas triunfan, otras son eliminadas.”

**Mao Tse Tung**

Clase potencial	Número de característica que se aplica
propietario	rechazada: 1 y 2 fallan, aunque la 6 aplica
sensor	aceptada: se aplican todas
panel de control	aceptada: se aplican todas
instalación	rechazada
sistema (alias sistema de seguridad)	aceptada: se aplican todas
número, tipo	rechazada: 3 fallan, atributos de sensores
password maestro	rechazada: 3 falla
número telefónico	rechazada: 3 falla
evento de sensor	aceptada: se aplican todas
alarma audible	aceptada: se aplican 2, 3, 4, 5 y 6
servicio de monitoreo	rechazada: 1 y 2 fallan aunque la 6 aplica

Debe notarse que: 1) la lista anterior no es exhaustiva; para completar el modelo tendrían que agregarse clases adicionales; 2) algunas de las clases potenciales rechazadas se convertirán en atributos para otras que sí fueron aceptadas (por ejemplo, *número* y *tipo* son atributos de **Sensor**, y *password*

*maestro y número telefónico* pueden convertirse en atributos de **Sistema**); 3) diferentes enunciados del problema harían que se tomaran decisiones distintas para “aceptar o rechazar” (por ejemplo, si cada propietario tuviera una clave individual o se identificara con reconocimiento de voz, la clase **Propietario** satisfaría las características 1 y 2, y se aceptaría).

### > 6.5.2 Especificación de atributos

**punto clave** Los atributos son el conjunto de objetos de datos que definen por completo la clase en el contexto del problema.

Los *atributos* describen una clase que ha sido seleccionada para incluirse en el modelo de requerimientos. En esencia, son los atributos los que definen la clase (esto aclara lo que significa la clase en el contexto del espacio del problema). Por ejemplo, si se fuera a construir un sistema que analiza estadísticas de jugadores de béisbol profesionales, los atributos de la clase **Jugador** serían muy distintos de los que tendría la misma clase cuando se usara en el contexto del sistema de pensiones de dicho deporte. En la primera, atributos tales como *nombre*, *porcentaje de bateo*, *porcentaje de fildeo*, *años jugados* y *juegos jugados* serían relevantes. Para la segunda, algunos de los anteriores sí serían significativos, pero otros se sustituirían (o se crearían) por atributos tales como *salario promedio*, *crédito hacia el retiro completo*, *opciones del plan de pensiones elegido*, *dirección de correo*, etcétera.

Para desarrollar un conjunto de atributos significativos para una clase de análisis, debe estudiarse cada caso de uso y seleccionar aquellas “cosas” que “pertenezcan” razonablemente a la clase. Además, debe responderse la pregunta siguiente para cada clase: “¿qué aspectos de los datos (compuestos o elementales) definen por completo esta clase en el contexto del problema en cuestión?”

Para ilustrarlo, se considera la clase **Sistema** definida para *CasaSegura*. El propietario configura la función de seguridad para que refleje la información de los sensores, la respuesta de la alarma, la activación o desactivación, la identificación, etc. Estos datos compuestos se representan del modo siguiente:

*información de identificación* = *identificación del sistema* + *número telefónico de verificación* + *estado del sistema*

*información de respuesta de la alarma* = *tiempo de retraso* + *número telefónico*

*información de activación o desactivación* = *password maestro* + *número de intentos permisibles* + *password temporal*

Cada uno de los datos a la derecha del signo igual podría definirse más, hasta un nivel elemental, pero para nuestros propósitos constituye una lista razonable de atributos para la clase **Sistema** (parte sombreada de la figura 6.9).

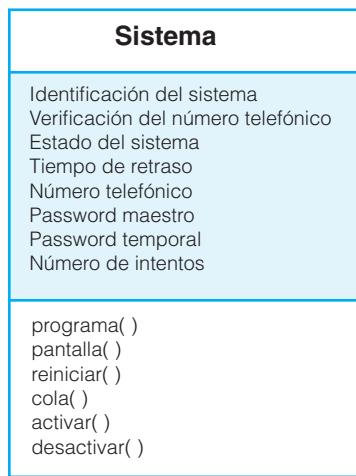
Los sensores forman parte del sistema general *CasaSegura*, pero no están enlistados como datos o atributos en la figura 6.9. **Sensor** ya se definió como clase, y se asociarán múltiples objetos **Sensor** con la clase **Sistema**. En general, se evita definir algo como atributo si más de uno va a asociarse con la clase.

### > 6.5.3 Definición de las operaciones

**consejo** Cuando se definen operaciones para una clase de análisis, hay que centrarse en el comportamiento orientado al problema y no en los comportamientos requeridos para su implementación.

Las *operaciones* definen el comportamiento de un objeto. Aunque existen muchos tipos distintos de operaciones, por lo general se dividen en cuatro categorías principales: 1) operaciones que manipulan datos en cierta manera (por ejemplo, los agregan, eliminan, editan, seleccionan, etc.), 2) operaciones que realizan un cálculo, 3) operaciones que preguntan sobre el estado de un objeto y 4) operaciones que vigilan un objeto en cuanto a la ocurrencia de un evento de control. Estas funciones se llevan a cabo con operaciones sobre los atributos o sobre asociaciones de éstos (véase la sección 6.5.5). Por tanto, una operación debe tener “conocimiento” de la naturaleza de los atributos y de las asociaciones de la clase.

Como primera iteración al obtener un conjunto de operaciones para una clase de análisis, se estudia otra vez una narración del procesamiento (o caso de uso) y se eligen aquellas que pertenezcan razonablemente a la clase. Para lograr esto, de nuevo se efectúa el análisis gramatical y se aislan los verbos. Algunos de éstos serán operaciones legítimas y se conectarán con facilidad a una clase específica. Por ejemplo, de la narración del procesamiento de *CasaSegura* ya presentada en este capítulo, se observa que “se asigna a sensor un número y tipo” o “se programa un password maestro para activar y desactivar el sistema” indican cierto número de cosas:



**FIGURA 6.4** Diagrama de clase para la clase sistema.

- Que una operación *asignar()* es relevante para la clase **Sensor**.
- Que se aplicará una operación *programar()* a la clase **Sistema**.
- Que *activar()* y *desactivar()* son operaciones que se aplican a la clase **Sistema**.

Hasta no hacer más investigaciones, es probable que la operación *programar()* se divida en cierto número de suboperaciones específicas adicionales que se requieren para configurar el sistema. Por ejemplo, *programar()* implica la especificación de números telefónicos, la configuración de las características del sistema (por ejemplo, crear la tabla de sensores, introducir las características de la alarma, etc.) y la introducción de la(s) clave(s). Pero, de momento, *programar()* se especifica como una sola operación.

Además del análisis gramatical, se obtiene más perspectiva sobre otras operaciones si se considera la comunicación que ocurre entre los objetos. Éstos se comunican con la transmisión de mensajes entre sí. Antes de continuar con la especificación de operaciones, se estudiará esto con más detalle.



## Modelos de clase

**La escena:** Cubículo de Ed, cuando comienza el modelado de los requerimientos.

**Participantes:** Jamie, Vinod y Ed, todos ellos miembros del equipo de ingeniería de software para *CasaSegura*.

### La conversación:

[Ed ha estado trabajando para obtener las clases a partir del formato del caso de uso para AVC-MVC (presentado en un recuadro anterior de este capítulo) y expone a sus colegas las que ha obtenido].

**Ed:** Entonces, cuando el propietario quiere escoger una cámara, la tiene que elegir del plano. Definí una clase llamada **Plano**. Éste es el diagrama. (Observan la figura 6.10.)

**Jamie:** Entonces, **Plano** es un objeto que agrupa paredes, puertas, ventanas y cámaras. Eso significa esas líneas con leyendas, ¿verdad?

**Ed:** Sí, se llaman “asociaciones”. Una clase se asocia con otra de acuerdo con las asociaciones que se ven (las asociaciones se estudian en la sección 6.5.5).

CasaSegura

**Vinod:** Es decir, el plano real está constituido por paredes que contienen en su interior cámaras y sensores. ¿Cómo sabe el plano dónde colocar estos objetos?

**Ed:** No lo sabe, pero las otras clases sí. Mira los atributos de, digamos, **SegmentodePared**, que se usa para construir una pared. El segmento de muro tiene coordenadas de inicio y final, y la operación `draw()` hace el resto.

**Jamie:** Y lo mismo vale para las ventanas y puertas. Parece como si cámara tuviera algunos atributos adicionales.

**Ed:** Sí. Los necesito para dar información del alcance y el acercamiento.

**Vinod:** Tengo una pregunta. ¿Por qué tiene la cámara una identificación pero las demás no? Veo que tienes un atributo llamado *ParedSigiente*. ¿Cómo sabe **SegmentodePared** cuál será la pared siguiente?

**Ed:** Buena pregunta, pero, como dijimos, ésa es una decisión de diseño, por lo que la voy a retrasar hasta...

**Jamie:** Momento... Apuesto a que ya lo has imaginado.

**Ed (sonríe con timidez):** Es cierto, voy a usar una estructura de lista que modelaré cuando vayamos a diseñar. Si somos puristas en cuanto a separar el análisis y el diseño, el nivel de detalle podría parecer sospechoso.

**Jamie:** Me parece muy bien, pero tengo más preguntas.

(Jamie hace preguntas que dan como resultado modificaciones menores.)

**Vinod:** ¿Tienes tarjetas CRC para cada uno de los objetos? Si así fuera, debemos actuar con ellas, sólo para estar seguros de que no hemos omitido nada.

**Ed:** No estoy seguro de cómo hacerlas.

**Vinod:** No es difícil y en verdad convienen. Te mostraré.

#### > 6.5.4 Modelado clase-responsabilidad-colaborador (CRC)

**cita:** "Un propósito de las tarjetas CRC es que fallen pronto, con frecuencia y en forma barata. Es mucho más barato desechar tarjetas que reorganizar una gran cantidad de código fuente."

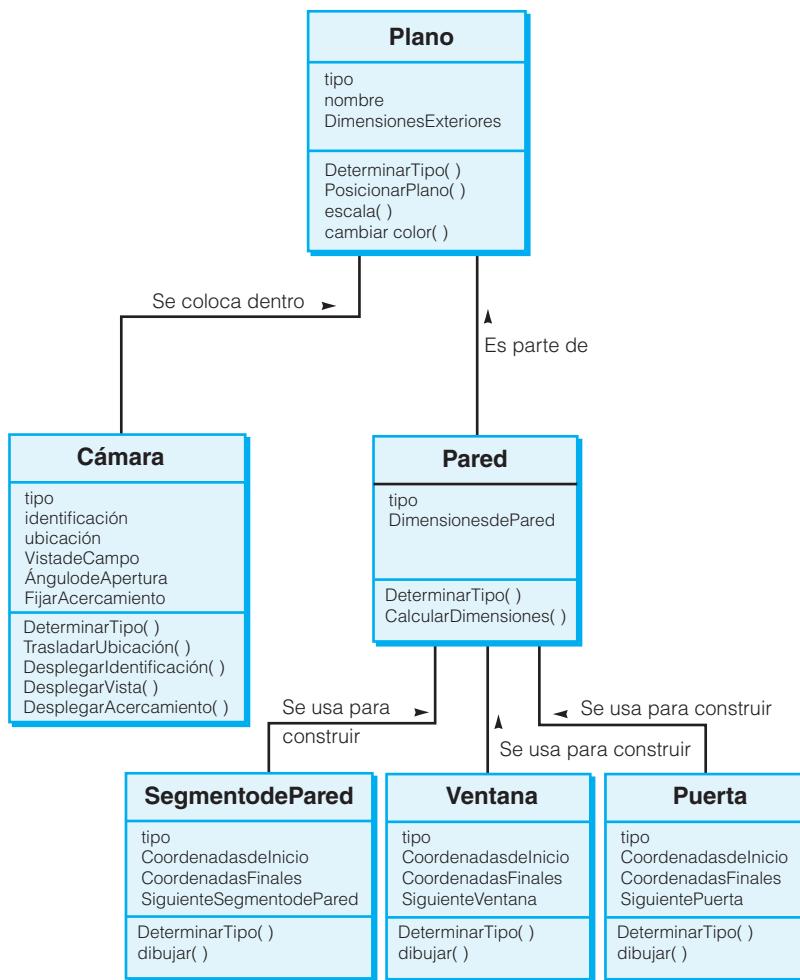
**C. Horstman**

El *modelado clase-responsabilidad-colaborador (CRC)* [Wir90] proporciona una manera sencilla de identificación y organización de las clases que son relevantes para los requerimientos de un sistema o producto. Ambler [Amb95] describe el modelado CRC en la siguiente forma:

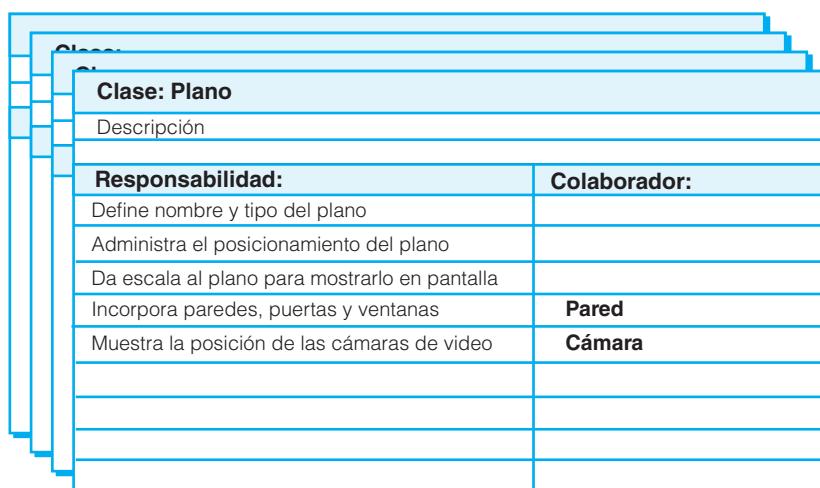
Un modelo CRC en realidad es un conjunto de tarjetas índice estándar que representan clases. Las tarjetas se dividen en tres secciones. En la parte superior de la tarjeta se escribe el nombre de la clase, en la parte izquierda del cuerpo se enlistan las responsabilidades de la clase y en la derecha, los colaboradores.

En realidad, el modelo CRC hace uso de tarjetas índice reales o virtuales. El objetivo es desarrollar una representación organizada de las clases. Las *responsabilidades* son los atributos y operaciones relevantes para la clase. En pocas palabras, una responsabilidad es "cualquier cosa que la clase sepa o haga" [Amb95]. Los *colaboradores* son aquellas clases que se requieren para dar a una clase la información necesaria a fin de completar una responsabilidad. En general, una *colaboración* implica una solicitud de información o de cierta acción.

En la figura 6.11 se ilustra una tarjeta CRC índice sencilla para la clase **Plano**: la lista de responsabilidades en la tarjeta CRC es preliminar y está sujeta a agregados o modificaciones. Las clases **Pared** y **Cámara** se anotan frente a la responsabilidad que requerirá su colaboración.



**FIGURA 6.10** Diagrama de clase para Plano (véase el análisis en el recuadro).



**FIGURA 6.11** Modelo de tarjeta CRC índice.

**WebRef** En la dirección [www.theumlcafe.com/a0079.htm](http://www.theumlcafe.com/a0079.htm) hay un análisis excelente de estos tipos de clase.

**cita:** "Pueden clasificarse científicamente los objetos en tres grandes categorías: los que no funcionan, los que se descomponen y los que se pierden."

**Russell Baker**

¿Qué lineamientos se aplican para asignar responsabilidades a las clases?

**Clases.** Al inicio de este capítulo se presentaron lineamientos básicos para identificar clases y objetos. La taxonomía de tipos de clase presentada en la sección 6.5.1 puede ampliarse con las siguientes categorías:

- *Clases de entidad*, también llamadas clases *modelo* o *de negocio*, se extraen directamente del enunciado del problema (por ejemplo, **Plano** y **Sensor**). Es común que estas clases representen cosas almacenadas en una base de datos y que persistan mientras dure la aplicación (a menos que se eliminen en forma específica).
- *Clases de frontera* se utilizan para crear la interfaz (por ejemplo, pantallas interactivas o reportes impresos) que el usuario mira y con la que interactúa cuando utiliza el software. Los objetos de entidad contienen información que es importante para los usuarios, pero no se muestran por sí mismos. Las clases de frontera se diseñan con la responsabilidad de administrar la forma en la que se presentan a los usuarios los objetos de entidad. Por ejemplo, una clase de frontera llamada **VentanadeCámara** tendría la responsabilidad de desplegar la salida de una cámara de vigilancia para el sistema *CasaSegura*.
- *Clases de controlador* administran una “unidad de trabajo” [UML03] de principio a fin. Es decir, las clases de controlador están diseñadas para administrar 1) la creación o actualización de objetos de entidad, 2) las instancias de los objetos de frontera en tanto obtienen información de los objetos de entidad, 3) la comunicación compleja entre conjuntos de objetos y 4) la validación de datos comunicados entre objetos o entre el usuario y la aplicación. En general, las clases de controlador no se consideran hasta haber comenzado la actividad de diseño.

**Responsabilidades.** En las secciones 6.5.2 y 6.5.3 se presentaron los lineamientos básicos para identificar responsabilidades (atributos y operaciones). Wirsfs-Brock *et al.* [Wir90] sugieren cinco lineamientos para asignar responsabilidades a las clases:

**1. La inteligencia del sistema debe estar distribuida entre las clases para enfrentar mejor las necesidades del problema.** Toda aplicación contiene cierto grado de inteligencia, es decir, lo que el sistema sabe y lo que puede hacer. Esta inteligencia se distribuye entre las clases de diferentes maneras. Las clases “tontas” (aquellas que tienen pocas responsabilidades) pueden modelarse para que actúen como subordinadas de ciertas clases “inteligentes” (las que tienen muchas responsabilidades). Aunque este enfoque hace directo el flujo del control en un sistema, tiene algunas desventajas: concentra toda la inteligencia en pocas clases, lo que hace que sea más difícil hacer cambios, y tiende a que se requieran más clases y por ello más trabajo de desarrollo.

Si la inteligencia del sistema tiene una distribución más pareja entre las clases de una aplicación, cada objeto sabe algo, sólo hace unas cuantas cosas (que por lo general están bien identificadas) y la cohesión del sistema mejora.<sup>17</sup> Esto facilita el mantenimiento del software y reduce el efecto de los resultados colaterales del cambio.

Para determinar si la inteligencia del sistema está distribuida en forma apropiada, deben evaluarse las responsabilidades anotadas en cada modelo de tarjeta CRC índice a fin de definir si alguna clase tiene una lista demasiado larga de responsabilidades. Esto indica una concentración de inteligencia.<sup>18</sup> Además, las responsabilidades de cada clase deben tener el mismo nivel de abstracción. Por ejemplo, entre las operaciones enlistadas para una clase agregada llamada **RevisarCuenta**, un revisor anota dos responsabilidades: *hacer el balance de la cuenta* y *eliminar comprobaciones concluidas*. La primera operación (responsabilidad) implica un procedimiento matemático complejo y lógico. La segunda es una simple actividad de oficina. Como estas dos operaciones no están en el mismo nivel de abstracción, *eliminar comprobaciones concluidas* debe colocarse dentro de las responsabilidades de **RevisarEntrada**, clase que está incluida en la clase agregada **RevisarCuenta**.

<sup>17</sup> La cohesión es un concepto de diseño que se estudia en el capítulo 8.

<sup>18</sup> En tales casos, puede ser necesario dividir la clase en una multiplicidad de ellas o completar subsistemas con el objeto de distribuir la inteligencia de un modo más eficaz.

**2. Cada responsabilidad debe enunciarse del modo más general posible.** Este lineamiento implica que las responsabilidades generales (tanto atributos como operaciones) deben residir en un nivel elevado de la jerarquía de clases (porque son generales y se aplicarán a todas las subclases).

**3. La información y el comportamiento relacionado con ella deben residir dentro de la misma clase.** Esto logra el principio orientado a objetos llamado *encapsulamiento*. Los datos y los procesos que los manipulan deben empacarse como una unidad cohesiva.

**4. La información sobre una cosa debe localizarse con una sola clase, y no distribuirse a través de muchas.** Una sola clase debe tener la responsabilidad de almacenar y manipular un tipo específico de información. En general, esta responsabilidad no debe ser compartida por varias clases. Si la información está distribuida, es más difícil dar mantenimiento al software y más complicado someterlo a prueba.

**5. Cuando sea apropiado, las responsabilidades deben compartirse entre clases relacionadas.** Hay muchos casos en los que varios objetos relacionados deben tener el mismo comportamiento al mismo tiempo. Por ejemplo, considere un juego de video que deba tener en la pantalla las clases siguientes: **Jugador**, **CuerpodelJugador**, **BrazosdelJugador**, **PiernasdelJugador** y **CabezadelJugador**. Cada una de estas clases tiene sus propios atributos (como *posición*, *orientación*, *color* y *velocidad*) y todas deben actualizarse y desplegarse a medida que el usuario manipula una palanca de juego. Las responsabilidades *actualizar()* y *desplegar()* deben, por tanto, ser compartidas por cada uno de los objetos mencionados. **Jugador** sabe cuando algo ha cambiado y requiere *actualizarse()*. Colabora con los demás objetos para obtener una nueva posición u orientación, pero cada objeto controla su propio despliegue en la pantalla.

**Colaboraciones.** Una clase cumple sus responsabilidades en una de dos formas: 1) usa sus propias operaciones para manipular sus propios atributos, con lo que satisface una responsabilidad particular o 2) colabora con otras clases. Wirfs-Brock *et al.* [Wir90] definen las colaboraciones del modo siguiente:

Las colaboraciones representan solicitudes que hace un cliente a un servidor para cumplir con sus responsabilidades. Una colaboración es la materialización del contrato entre el cliente y el servidor [...] Decimos que un objeto colabora con otro si, para cumplir una responsabilidad, necesita enviar al otro objeto cualesquier mensajes. Una sola colaboración fluye en una dirección: representa una solicitud del cliente al servidor. Desde el punto de vista del cliente, cada una de sus colaboraciones está asociada con una responsabilidad particular implementada por el servidor.

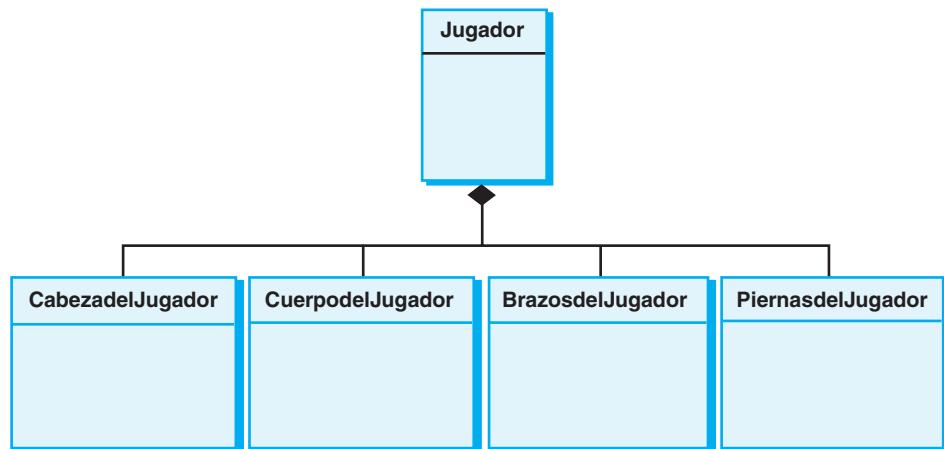
Las colaboraciones se identifican determinando si una clase puede cumplir cada responsabilidad. Si no es así, entonces necesita interactuar con otra clase. Ésa es una colaboración.

Como ejemplo, considere la función de seguridad de *CasaSegura*. Como parte del procedimiento de activación, el objeto **PaneldeControl** debe determinar si están abiertos algunos sensores. Se define una responsabilidad llamada *determinar-estado-delsensor()*. Si los sensores están abiertos, **PaneldeControl** debe fijar el atributo *estado* como "no está listo". La información del sensor se adquiere de cada objeto **Sensor**. Por tanto, la responsabilidad *determinar-estado-delsensor()* se cumple sólo si **PaneldeControl** trabaja en colaboración con **Sensor**.

Para ayudar a identificar a los colaboradores, se estudian tres relaciones generales diferentes entre las clases [Wir90]: 1) la relación *es-parte-de*, 2) la relación *tiene-conocimiento-de* y 3) la relación *depende-de*. En los párrafos siguientes se analizan brevemente cada una de estas tres responsabilidades generales.

Todas las clases que forman parte de una clase agregada se conectan a ésta por medio de una relación *es-parte-de*. Considere las clases definidas por el juego mencionado antes, la clase **CuerpodelJugador** *es-parte-de* **Jugador**, igual que **BrazosdelJugador**, **PiernasdelJugador** y **CabezadelJugador**. En UML, estas relaciones se representan como el agregado que se ilustra en la figura 6.12.

Cuando una clase debe adquirir información de otra, se establece la relación *tiene-conocimiento-de*. La responsabilidad *determinar-estado-delsensor()* ya mencionada es un ejemplo de ello.



**FIGURA 6.12** Una clase agregada compuesta.

La relación *depende-de* significa que dos clases tienen una dependencia que no se determina por *tiene-conocimiento-de* ni por *es-partde-de*. Por ejemplo, **CabezadelJugador** siempre debe estar conectada a **CuerpodelJugador** (a menos que el juego de video sea particularmente violento), pero cada objeto puede existir sin el conocimiento directo del otro. Un atributo del objeto **CabezadelJugador**, llamado *posición-central*, se determina a partir de la posición central de **CuerpodelJugador**. Esta información se obtiene por medio de un tercer objeto, **Jugador**, que la obtiene de **CuerpodelJugador**. Entonces, **CabezadelJugador** *depende-de* **CuerpodelJugador**.

En todos los casos, el nombre de la clase colaboradora se registra en el modelo de tarjeta CRC índice, junto a la responsabilidad que produce la colaboración. Por tanto, la tarjeta índice contiene una lista de responsabilidades y las colaboraciones correspondientes que hacen que se cumplan (véase la figura 6.11).

Cuando se ha desarrollado un modelo CRC completo, los participantes lo revisan con el empleo del enfoque siguiente [Amb95]:

1. Se da a todos los participantes que intervienen en la revisión (del modelo CRC) un subconjunto del modelo de tarjetas índice CRC. Deben separarse aquellas que colaboran (de modo que ningún revisor deba tener dos tarjetas que colaboren).
2. Todos los escenarios de casos de uso (y los diagramas correspondientes) deben organizarse en dos categorías.
3. El líder de la revisión lee el caso de uso en forma deliberada. Cuando llega a un objeto con nombre, entrega una ficha a la persona que tenga la tarjeta índice de la clase correspondiente. Por ejemplo, un caso de uso de *CasaSegura* contiene la narración siguiente:

El propietario observa el panel de control de *CasaSegura* para determinar si el sistema está listo para recibir una entrada. Si el sistema no está listo, el propietario debe cerrar físicamente las puertas y ventanas de modo que el indicador *listo* aparezca [un indicador *no está listo* implica que un sensor se encuentra abierto, es decir, que una puerta o ventana está abierta].

Cuando en la narración del caso de uso el líder de la revisión llega a “panel de control”, entrega la ficha a la persona que tiene la tarjeta índice **PaneldeControl**. La frase “implica que un sensor está abierto” requiere que la tarjeta índice contenga una responsabilidad que validará esta implicación (esto lo logra la responsabilidad *determinar-estado-delsensor()*). Junto a la responsabilidad, en la tarjeta índice se encuentra el **Sensor** colaborador. Entonces, la ficha pasa al objeto **Sensor**.

4. Cuando se pasa la ficha, se pide al poseedor de la tarjeta **Sensor** que describa las responsabilidades anotadas en la tarjeta. El grupo determina si una (o más) de las responsabilidades satisfacen el requerimiento del caso de uso.

5. Si las responsabilidades y colaboraciones anotadas en las tarjetas índice no se acomodan al caso de uso, éstas se modifican. Lo anterior tal vez incluya la definición de nuevas clases (y las tarjetas CRC índice correspondientes) o la especificación en las tarjetas existentes de responsabilidades o colaboraciones nuevas o revisadas.

Este modo de operar continúa hasta terminar el caso de uso. Cuando se han revisado todos los casos de uso, continúa el modelado de los requerimientos.



## Modelos CRC

CasaSegura

**La escena:** Cubículo de Ed cuando comienza el modelado de los requerimientos.

**Participantes:** Vinod y Ed, miembros del equipo de ingeniería de software de *CasaSegura*.

### La conversación:

[Vinod ha decidido enseñar a Ed con un ejemplo cómo desarrollar las tarjetas CRC.]

**Vinod:** Mientras tú trabajabas en la vigilancia y Jamie lo hacía con la seguridad, yo estaba en la función de administración del hogar.

**Ed:** ¿Cuál es el estado de eso? Mercadotecnia cambia lo que quiere a cada rato.

**Vinod:** Aquí está la primera versión de caso de uso para toda la función... la mejoramos un poco, pero debe darte el panorama general...

**Caso de uso:** Función de administración de *CasaSegura*.

**Narración:** Queremos usar la interfaz de administración del hogar en una PC o en una conexión de internet para controlar los dispositivos electrónicos que tengan controladores de interfaz inalámbrica. El sistema debe permitir encender y apagar focos específicos, controlar aparatos conectados a una interfaz inalámbrica y fijar el sistema de calefacción y aire acondicionado a la temperatura que desee. Para hacer esto, quiero seleccionar los aparatos en el plano de la casa. Cada equipo debe estar identificado en el plano. Como característica opcional, quiero controlar todos los equipos audiovisuales: sonido, televisión, DVD, grabadoras digitales, etcétera.

Con una sola selección, quiero preparar toda la casa para distintas situaciones. Una es *casa*, otra es *salir*, la tercera es *viaje nocturno* y la cuarta es *viaje largo*. Todas estas situaciones tienen especificaciones que se aplicarán a todos los equipos. En los estados de *viaje nocturno* y *viaje largo*, el sistema debe encender y apagar focos en momentos elegidos al azar (para que parezca que hay alguien en casa) y controlar el sistema de calefacción y aire acondicionado. Debo poder hacer esta preparación por internet, con la protección de claves adecuadas...

**Ed:** ¿El personal de hardware ya tiene listas todas las interfaces inalámbricas?

**Vinod (sonríe):** Están trabajando en eso; dicen que no hay problema. De cualquier forma, obtuve muchas clases para la administración del hogar y podemos usar una como ejemplo. Tomemos la clase **InterfazdeAdministracióndelHogar**.

**Ed:** Bien... entonces, las responsabilidades son... los atributos y operaciones para la clase, y las colaboraciones son las clases que indican las responsabilidades.

**Vinod:** Pensé que no habías entendido el concepto CRC.

**Ed:** Un poco, quizás, pero continúa.

**Vinod:** Aquí está mi definición de la clase **InterfazdeAdministracióndelHogar**.

### Atributos:

PaneldeOpciones: contiene información sobre los botones que permiten al usuario seleccionar funcionalidad.

**PaneldeSituación:** contiene información acerca de los botones que permiten que el usuario seleccione la situación.

**Plano:** igual que el objeto de vigilancia, pero éste muestra los equipos.

**ÍconosdeAparatos:** informa sobre los íconos que representan luces, aparatos, calefacción y aire acondicionado, etcétera.

**PaneldeAparatos:** simula el panel de control de un aparato o equipo; permite controlarlo.

#### Operaciones:

*DesplegarControl( ), SeleccionarControl( ), DesplegarSituación( ), SeleccionarSituación( ), AccederaPlano( ), SeleccionarÍcono deEquipo( ), DesplegarPaneldeEquipo( ), AccederaPaneldeEquipo( ),...*

**Clase:** InterfazdeAdministracióndelHogar

#### Responsabilidad Colaborador

*DesplegarControl( ) PaneldeOpciones (clase)*

*SeleccionarControl( ) PaneldeOpciones (clase)*

*DesplegarSituación( ) PaneldeSituación (clase)*

*SeleccionarSituación( ) PaneldeSituación (clase)*

*AccederaPlano( ) Plano (clase) . . .*

...

**Ed:** De modo que cuando se invoque a operación *AccederaPlano( )*, colabora con el objeto **Plano** de igual manera que el que desarrollamos para vigilancia. Espera, aquí tengo su descripción (ven la figura 6.10).

**Vinod:** Exactamente. Y si quisieramos revisar todo el modelo de la clase, podríamos comenzar con esta tarjeta índice, luego iríamos a la del colaborador y de ahí a una de los colaboradores del colaborador, y así sucesivamente.

**Ed:** Buena forma de encontrar omisiones o errores.

**Vinod:** Sí.

### > 6.5.5 Asociaciones y dependencias

**punto clave** Una asociación define una relación entre clases. La multiplicidad define cuántas de una clase se relacionan con cuántas de otra clase.

¿Qué es un estereotipo?

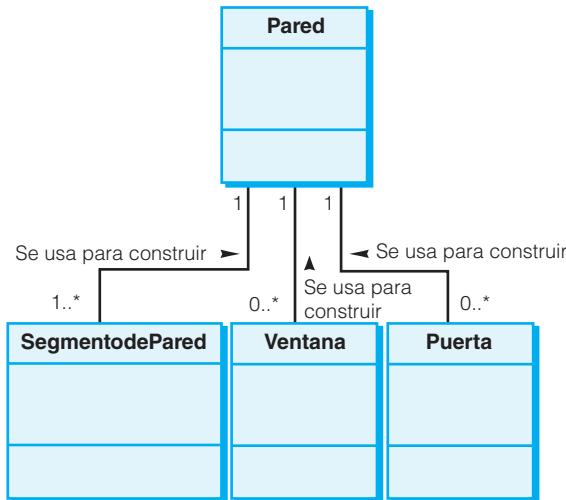
En muchos casos, dos clases de análisis se relacionan de cierto modo con otra, en forma muy parecida a como dos objetos de datos se relacionan entre sí (véase la sección 6.4.3). En UML, estas relaciones se llaman *asociaciones*. Al consultar la figura 6.10, la clase **Plano** se define con la identificación de un conjunto de asociaciones entre **Plano** y otras dos clases, **Cámara** y **Pared**. La clase **Pared** está asociada con tres clases que permiten que se construya ésta, y que son **SegmentodePared**, **Ventana** y **Puerta**.

En ciertos casos, una asociación puede definirse con más detalle si se indica *multiplicidad*. En relación con la figura 6.10, un objeto **Pared** se construye a partir de uno o más objetos **SegmentodePared**. Además, el objeto **Pared** puede contener 0 o más objetos **Ventana** y 0 o más objetos **Puerta**. Estas restricciones de multiplicidad se ilustran en la figura 6.13, donde “uno o más” se representa con  $1\dots^*$ , y para “0 o más” se usa  $0\dots^*$ . En LMU, el asterisco indica una frontera ilimitada en ese rango.<sup>19</sup>

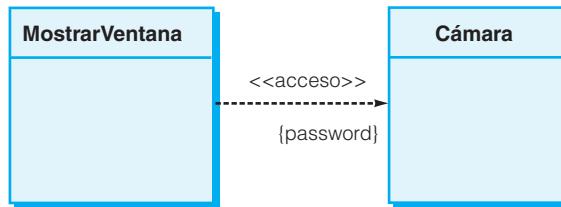
Sucede con frecuencia que entre dos clases de análisis existe una relación cliente-servidor. En tales casos, una clase cliente depende de algún modo de la clase servidor, y se establece una *relación de dependencia*. Las dependencias están definidas por un estereotipo. Un *estereotipo* es un “mecanismo extensible” [Arl02] dentro del UML que permite definir un elemento especial de modelado con semántica y especialización determinadas. En UML, los estereotipos se representan entre paréntesis dobles angulares (por ejemplo, <<estereotipo>>).

<sup>19</sup> Como parte de una asociación, pueden indicarse otras relaciones de multiplicidad: una a una, una a muchas, muchas a muchas, una a un rango específico con límites inferior y superior, y otras.

Como ilustración de una dependencia simple dentro del sistema de vigilancia *CasaSegura*, un objeto **Cámara** (la clase servidora, en este caso) proporciona una imagen a un objeto **MostrarVentana** (la clase cliente). La relación entre estos dos objetos no es una asociación simple sino de dependencia. En el caso de uso escrito para la vigilancia (que no se presenta aquí), debe darse una clave especial a fin de observar ubicaciones específicas de las cámaras. Una forma de lograr esto es hacer que **Cámara** pida un password y luego asegure el permiso a **MostrarVentana** para que presente el video. Esto se representa en la figura 6.14, donde <>acceso>> implica que el uso de la salida de cámara se controla con una clave especial.



**FIGURA 6.13** Multiplicidad.



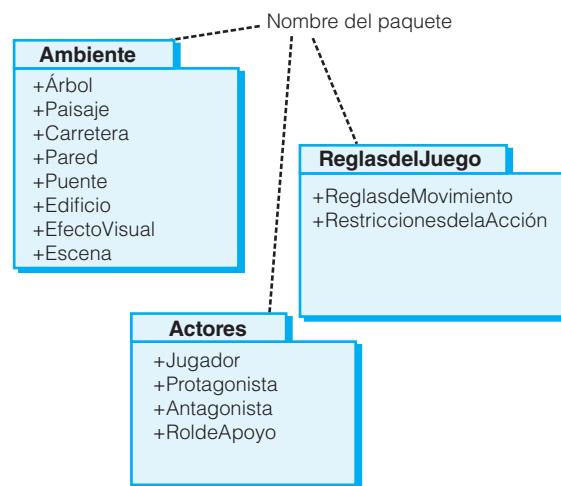
**FIGURA 6.14** Dependencias.

### > 6.5.6 Paquetes de análisis

Una parte importante del modelado del análisis es la categorización. Es decir, se clasifican distintos elementos del modelo de análisis (por ejemplo, casos de uso y clases de análisis) de manera que se agrupen en un paquete —llamado *paquete de análisis*— al que se da un nombre representativo.

Para ilustrar el uso de los paquetes de análisis, considere el juego de video que se mencionó antes. A medida que se desarrolla el modelo de análisis para el juego de video, se obtiene un gran número de clases. Algunas se centran en el ambiente del juego —las escenas visuales que el usuario ve cuando lo usa—. En esta categoría quedan clases tales como **Árbol**, **Paisaje**, **Carretera**, **Pared**, **Puente**, **Edificio** y **EfectoVisual**. Otras se centran en los caracteres dentro del juego y describen sus características físicas, acciones y restricciones. Pueden definirse clases como **Jugador** (ya descrita), **Protagonista**, **Antagonista** y **RolesdeApoyo**. Otras más describen las reglas del juego —cómo se desplaza un jugador por el ambiente—. Candidatas para esto son clases como **ReglasdeMovimiento** y **RestriccionesdelaAcción**. Pueden existir muchas otras categorías. Estas clases se agrupan en los paquetes de análisis que se observan en la figura 6.15.

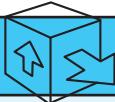
**punto clave** Un paquete se utiliza para ensamblar un conjunto de clases relacionadas



**FIGURA 6.12** Paquetes.

El signo *más* (suma) que precede al nombre de la clase de análisis en cada paquete, indica que las clases tienen visibilidad pública, por lo que son accesibles desde otros paquetes. Aunque no se aprecia en la figura, hay otros símbolos que preceden a un elemento dentro de un paquete. El signo *menos* (resta) indica que un elemento queda oculto desde todos los demás paquetes. Y el símbolo # señala que un elemento es accesible sólo para los paquetes contenidos dentro de un paquete dado.

## Resumen



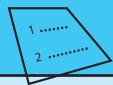
El objetivo del modelado de los requerimientos es crear varias representaciones que describan lo que necesita el cliente, establecer una base para generar un diseño de software y definir un conjunto de requerimientos que puedan ser validados una vez construido el software. El modelo de requerimientos cruza la brecha entre la representación del sistema que describe el sistema en su conjunto y la funcionalidad del negocio, y un diseño de software que describe la arquitectura de la aplicación del software, la interfaz de usuario y la estructura de componentes.

Los modelos basados en el escenario ilustran los requerimientos del software desde el punto de vista del usuario. El caso de uso —descripción, hecha con una narración o un formato, de una interacción entre un actor y el software— es el principal elemento del modelado. El caso de uso se obtiene durante la indagación de los requerimientos y define las etapas clave de una función o interacción específica. El grado de formalidad del caso de uso y su nivel de detalle varía, pero el resultado final da las entradas necesarias a todas las demás actividades del modelado. Los escenarios también pueden ser descritos con el uso de un diagrama de actividades —representación gráfica parecida a un diagrama de flujo que ilustra el flujo del

procesamiento dentro de un escenario específico—. Los diagramas de canal (swimlane) ilustran la forma en la que se asigna el flujo del procesamiento a distintos actores o clases.

El modelado de datos se utiliza para describir el espacio de información que será construido o manipulado por el software. El modelado de datos comienza con la representación de los objetos de datos —información compuesta que debe ser entendida por el software—. Se identifican los atributos de cada objeto de datos y se describen las relaciones entre estos objetos.

El modelado basado en clases utiliza información obtenida de los elementos del modelado basado en el escenario y en datos, para identificar las clases de análisis. Se emplea un análisis gramatical para obtener candidatas a clase, atributos y operaciones, a partir de narraciones basadas en texto. Se definen criterios para definir una clase. Para definir las relaciones entre clases, se emplean tarjetas índice clase-responsabilidad-colaborador. Además, se aplican varios elementos de la notación UML para definir jerarquías, relaciones, asociaciones, agregaciones y dependencias entre clases. Se emplean paquetes de análisis para clasificar y agrupar clases, de manera que sean más manejables en sistemas grandes.



## Problemas y puntos por evaluar

**6.1.** ¿Es posible comenzar a codificar de inmediato después de haber creado un modelo de análisis? Explique su respuesta y luego defienda el punto de vista contrario.

**6.2.** Una regla práctica del análisis es que el modelo “debe centrarse en los requerimientos visibles dentro del dominio del problema o negocio”. ¿Qué tipos de requerimientos *no* son visibles en dichos dominios? Dé algunos ejemplos.

**6.3.** ¿Cuál es el propósito del análisis del dominio? ¿Cómo se relaciona con el concepto de patrones de requerimientos?

**6.4.** ¿Es posible desarrollar un modelo de análisis eficaz sin desarrollar los cuatro elementos que aparecen en la figura 6.3? Explique su respuesta.

**6.5.** Se pide al lector que construya uno de los siguientes sistemas:

- a) Sistema de inscripción a la universidad basado en red.
- b) Sistema de procesamiento de órdenes basado en web para una tienda de computadoras.
- c) Sistema de facturación simple para un negocio pequeño.
- d) Libro de cocina basado en internet, construido en un horno eléctrico o de microondas.

Seleccione el sistema que le interese y desarrolle un diagrama entidad-relación que describa los objetos de datos, relaciones y atributos.

**6.6.** El departamento de obras públicas de una gran ciudad ha decidido desarrollar un sistema de seguimiento y reparación de baches, basado en web (SSRB).

Cuando se reportan los baches, se registran en “sistema de reparación del departamento de obras públicas” y se les asigna un número de identificación, almacenado según la calle, tamaño (en una escala de 1 a 10), ubicación (en medio, cuneta, etc.),

distrito (se determina con la dirección en la calle) y prioridad de reparación (determinada por el tamaño del bache). Los datos de la orden de trabajo se asocian con cada bache e incluyen su ubicación y tamaño, número de identificación del equipo de reparación, número de personas en dicho equipo, equipo asignado, horas dedicadas a la reparación, estado del bache (trabajo en proceso, reparado, reparación temporal, no reparado), cantidad de material de relleno utilizado y costo de la reparación (calculado a partir de las horas dedicadas, número de personas, materiales y equipo empleado). Por último, se crea un archivo de daños para mantener la información sobre daños reportados debido al bache, y se incluye el nombre y dirección del ciudadano, número telefónico, tipo de daño y cantidad de dinero por el daño. El SSRB es un sistema en línea, todas las solicitudes se harán en forma interactiva.

- a) Dibuje un diagrama UML para el caso de uso del sistema SSRB. Tendrá que hacer algunas suposiciones sobre la manera en la que un usuario interactúa con el sistema.
- b) Desarrolle un modelo de clase para el sistema SSRB.

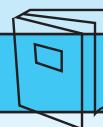
**6.7.** Escriba un caso de uso basado en formato para el sistema de administración del hogar *CasaSegura* descrito de manera informal en el recuadro de la sección 6.5.4.

**6.8.** Desarrolle un conjunto completo de tarjetas índice de modelo CRC, sobre el producto o sistema que elija como parte del problema 6.5.

**6.9.** Revise con sus compañeros las tarjetas índice CRC. ¿Cuántas clases, responsabilidades y colaboradores adicionales fueron agregados como consecuencia de la revisión?

**6.10.** ¿Qué es y cómo se usa un paquete de análisis?

## Lecturas adicionales y fuentes de información



Los casos de uso son el fundamento de todos los enfoques del modelado de los requerimientos. El tema se analiza con amplitud en Rosenberg y Stephens (*Use Case Driven Object Modeling with UML: Theory and Practice*, Apress, 2007), Denny (*Succeeding with Use Cases: Working Smart to Deliver Quality*, Addison-Wesley, 2005), Alexander y Maiden (eds.) (*Scenarios, Stories, Use Cases: Through the Systems Development Life-Cycle*, Wiley, 2004), Bittner y Spence (*Use Case Modeling*, Addison-Wesley, 2002), Cockburn [Coc01b] y en otras referencias mencionadas en los capítulos 5 y 6.

El modelado de datos constituye un método útil para examinar el espacio de información. Los libros de Hoberman [Hob06] y Simsion y Witt [Sim05] hacen tratamientos razonablemente am-

plios. Además, Allen y Terry (*Beginning Relational Data Modeling*, 2a. ed., Apress, 2005), Allen (*Data Modeling for Everyone*, Word Press, 2002), Teorey *et al.* (*Database Modeling and Design: Logical Design*, 4a. ed., Morgan Kaufmann, 2005) y Carlis y Maguire (*Mastering Data Modeling*, Addison-Wesley, 2000) presentan métodos de aprendizaje detallados para crear modelos de datos de calidad industrial. Un libro interesante escrito por Hay (*Data Modeling Patterns*, Dorset House, 1995) presenta patrones comunes de modelos de datos que se encuentran en muchos negocios diferentes.

Análisis de técnicas de modelado UML que pueden aplicarse tanto para el análisis como para el diseño se encuentran en O'Docherty (*Object-Oriented Analysis and Design: Understanding System Development with UML 2.0*, Wiley, 2005), Arlow y Neustadt

(*UML, 2 and the Unified Process*, 2a. ed., Addison-Wesley, 2005), Roques (*UML in Practice*, Wiley, 2004), Dennis *et al.* (*Systems Analysis and Design with UML Version 2.0*, Wiley, 2004), Larman (*Applying UML and Patterns*, 2a. ed., Prentice-Hall, 2001) y Rosenberg y Scott (*Use Case Driven Object Modeling with UML*, Addison-Wesley, 1999).

En internet existe una amplia variedad de fuentes de información sobre el modelado de requerimientos. En el sitio web del libro, en [www.mhhe.com/engcs/comsci/pressman/professioal/olc/ser.htm](http://www.mhhe.com/engcs/comsci/pressman/professioal/olc/ser.htm), se halla una lista actualizada de referencias en web que son relevantes para el modelado del análisis.