4.1. Presentación del capítulo

En este capítulo tratamos los fundamentos básicos del modelado de caso de uso, que es otra forma de ingeniería de requisitos. Le mostramos el proceso del modelado de caso de uso según se define por UP. Nos centramos en técnicas específicas y estrategias que los analistas/diseñadores orientados a objetos pueden utilizar para realizar de forma eficaz el modelado de caso de uso. Para centrarse en estas técnicas, mantenemos los casos de uso en este apartado lo más sencillo posible. Existe un ejemplo completo (y más complejo) en nuestro sitio Web en la dirección `www.umlandtheunifiedprocess.com`.

UML no especifica ninguna estructura formal para la especificación de caso de uso. Esto es problemático, ya que diferentes modeladores adoptan estándares diferentes. Para ayudar con esto, hemos adoptado un estándar sencillo y eficaz en este capítulo y en nuestro ejemplo. Para ayudarle a aplicar nuestro enfoque, nuestro sitio Web proporciona esquema XML (*eXtensible Markup Language*) de código abierto para casos de uso y actores que puede utilizar libremente en sus proyectos. Estas plantillas están basadas en las mejores prácticas de la industria y proporcionan un estándar sencillo para capturar especificaciones de caso de uso.

Nuestro sitio Web también incluye una sencilla hoja de estilo XSL (*eXtensible Stylesheet Language*) que transforma los documentos XML de caso de uso en HTML para mostrarse en un navegador. Esta hoja de estilo es un ejemplo de utilidad que se puede personalizar fácilmente para incorporar otros estándares de documento

para diferentes organizaciones. Una explicación detallada de XML va más allá del alcance de este libro y debería consultar textos de XML como [Pitts 1] y [Kay 1] para utilizar estos documentos de forma eficaz.

Aparte del esquema de código abierto y hojas de estilo, estamos trabajando en un enfoque más flexible denominado SUMR (*Simple Use case Markup Restructured*). Se trata de un sencillo lenguaje de codificación de texto estructurado de código abierto para casos de uso y actores. Proporcionamos esquemas SUMR de ejemplo, analizadores y generadores XML y HTML en nuestro sitio Web.

4.2. Modelado de caso de uso

El modelado de caso de uso es una forma de ingeniería de requisitos. En el capítulo 3 vio cómo crear un modelo de requisitos que englobaba requisitos funcionales y no funcionales en lo que podríamos denominar la forma "tradicional". El modelado de caso de uso es una forma diferente y complementaria de crear y documentar requisitos. El modelado de casos de uso funciona de la siguiente forma:

- Encontrar el límite de un sistema candidato.
- Encontrar los actores.
- Encontrar los casos de uso:
  - Especificar el caso de uso.
  - Identificar los flujos alternativos clave.
- Pasar en bucle hasta que los casos de uso, actores y límite del sistema son estables.

Generalmente, empieza con una estimación inicial de dónde se encuentra el límite del sistema para ayudarle a definir el ámbito de aplicación de la actividad de modelado. Las acciones se realizan luego iterativamente y a menudo en paralelo.

El resultado de estas actividades es el modelo de caso de uso. Existen cuatro componentes de este modelo.

- **Límite del sistema:** Un cuadro dibujado alrededor de los casos de uso para indicar el borde o límite del sistema que se modela. Esto se conoce como el sujeto en UML 2.
- **Actores:** Roles desempeñados por personas o elementos que utilizan el sistema.
- **Casos de uso:** Lo que los actores pueden hacer con el sistema.
- **Relaciones:** Relaciones significativas entre los actores y casos de uso.

El modelo de caso de uso proporciona una fuente principal para objetos y clases. Es la entrada principal para el modelado de clases.

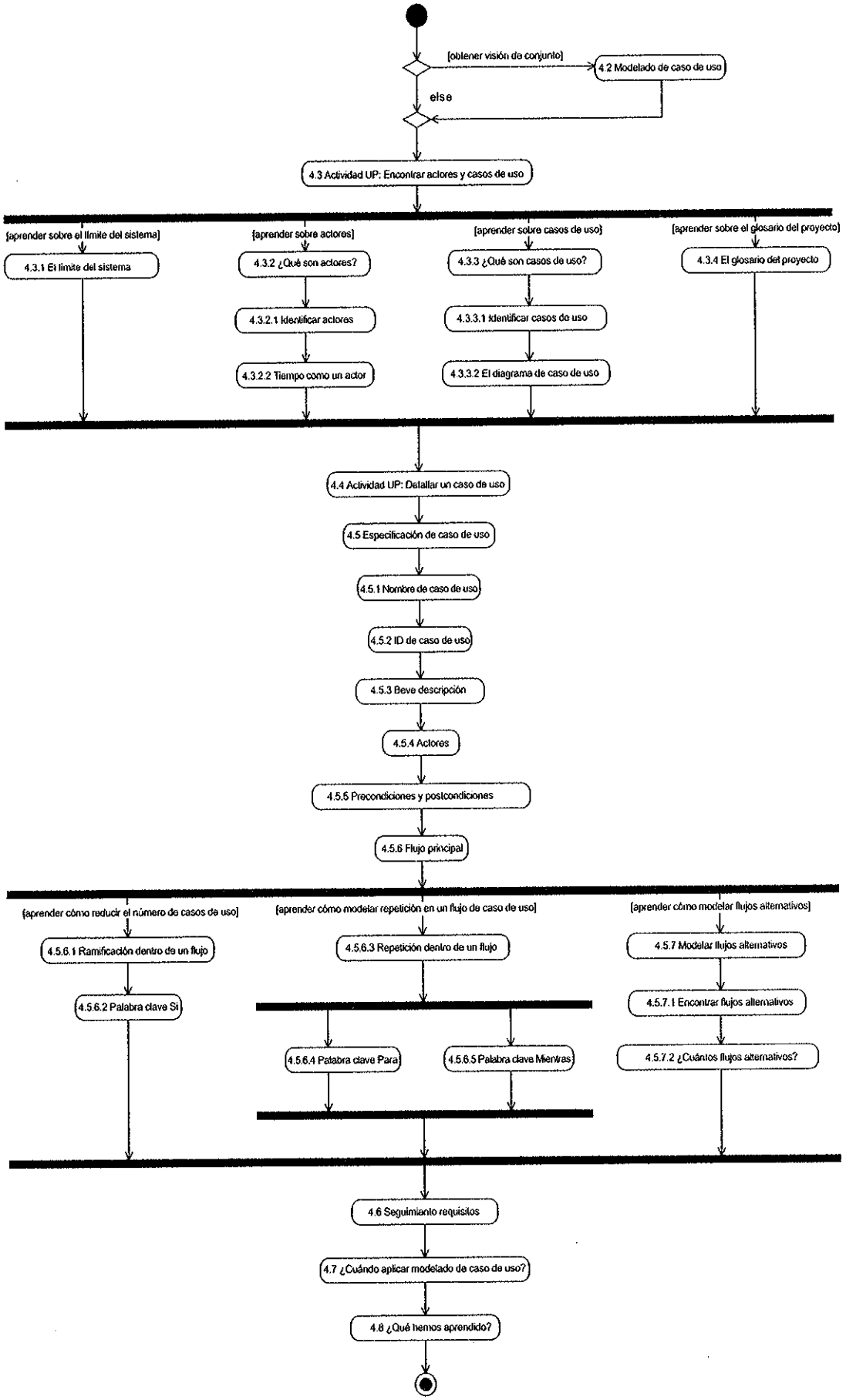

[obtener visión de conjunto]
4.2 Modelado de caso de uso
else
4.3 Actividad UP: Encontrar actores y casos de uso
[aprender sobre el límite del sistema]
[aprender sobre actores]
[aprender sobre casos de uso]
[aprender sobre el glosario del proyecto]
4.3.1 El límite del sistema
4.3.2 ¿Qué son actores?
4.3.3 ¿Qué son casos de uso?
4.3.4 El glosario del proyecto
4.3.2.1 Identificar actores
4.3.3.1 Identificar casos de uso
4.3.2.2 Tiempo como un actor
4.3.3.2 El diagrama de caso de uso
4.4 Actividad UP: Detallar un caso de uso
4.5 Especificación de caso de uso
4.5.1 Nombre de caso de uso
4.5.2 ID de caso de uso
4.5.3 Breve descripción
4.5.4 Actores
4.5.5 Precondiciones y postcondiciones
4.5.6 Flujo principal
[aprender cómo reducir el número de casos de uso]
[aprender cómo modelar repetición en un flujo de caso de uso]
[aprender cómo modelar flujos alternativos]
4.5.6.1 Ramificación dentro de un flujo
4.5.6.3 Repetición dentro de un flujo
4.5.7 Modelar flujos alternativos
4.5.6.2 Palabra clave Si
4.5.7.1 Encontrar flujos alternativos
4.5.6.4 Palabra clave Para
4.5.6.5 Palabra clave Mientras
4.5.7.2 ¿Cuántos flujos alternativos?
4.6 Seguimiento requisitos
4.7 ¿Cuándo aplicar modelado de caso de uso?
4.8 ¿Qué hemos aprendido?

Figura 4.1.

4.3. Actividad UP: encontrar actores y casos de uso

En este apartado nos centramos en la actividad Encontrar actores y casos de uso del workflow de requisitos (consulte el capítulo 3). Esto se muestra en la figura 4.2. Continuaremos examinando la actividad `Detallar un caso de uso`.

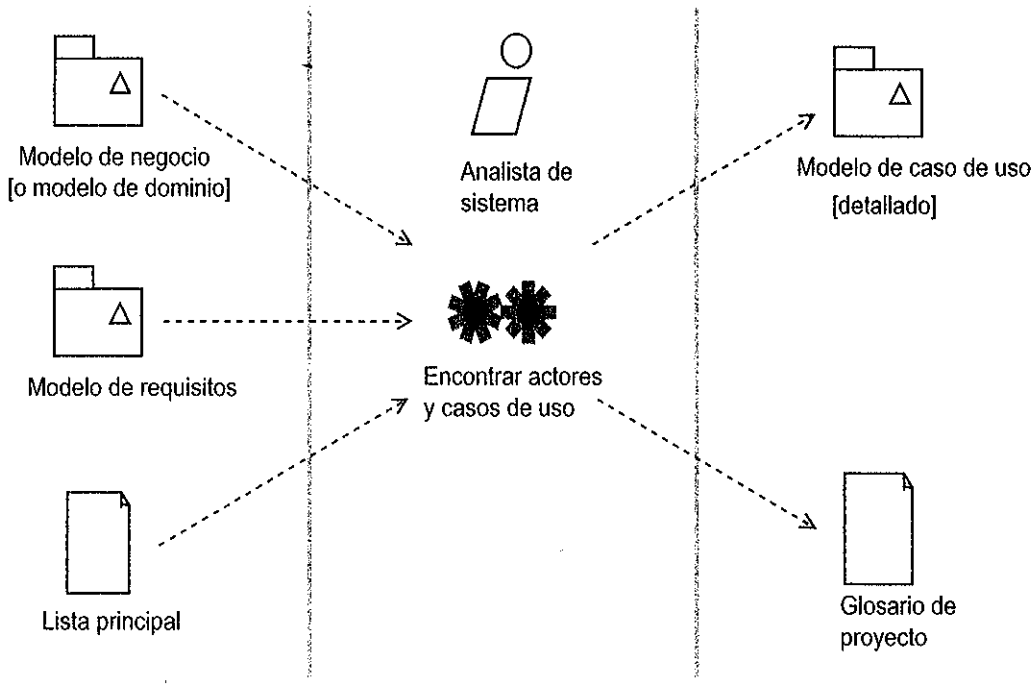


Figura 4.2. Adaptada de la figura 7.11 [Jacobson 1] con permiso de Addison-Wesley.

Merece la pena mirar las entradas para Encontrar actores y casos de uso:

- **Modelo de negocio:** Puede tener o no un modelo de negocio disponible que esté relacionado con el sistema que esté modelando. Si lo tiene, ésta es una excelente fuente de requisitos.
- **Modelo de requisitos:** Describimos la creación de este modelo en el capítulo 3. Estos requisitos proporcionan entrada de utilidad para el proceso de modelado de caso de uso. En particular, los requisitos funcionales sugerirán casos de uso y actores. Los requisitos no funcionales sugerirán algo que debería mantener en mente cuando construye el modelo de caso de uso.
- **Lista principal:** Éste es un conjunto de requisitos candidatos que pueden adoptar la forma de un documento de visión o similar.

En el trabajo original de Jacobson, el modelo de requisitos (mostrado en gris en la figura 4.2 para mostrar que está modificado de la figura original) se reemplazó por requisitos adicionales. Este documento constaba de requisitos (normalmente no funcionales) que no estaban relacionados con ningún caso de uso en particular.

El documento de requisitos adicionales fue inicialmente algo general para requisitos no funcionales que atraviesa los casos de uso. En nuestro enfoque mucho más robusto para la ingeniería de requisitos es un subconjunto del modelo de requisitos.

4.3.1. El sujeto (límite del sistema)

Lo primero que necesita hacer cuando piense en crear un sistema es decidir dónde se encuentran los límites del sistema. Es decir, necesita decidir qué es parte de su sistema (dentro de los límites del sistema) y qué es externo a su sistema (fuera de los límites del sistema). Esto suena obvio, pero hemos encontrado muchos proyectos donde han surgido problemas serios de unos límites del sistema dudosos. La posición del límite del sistema tiene un impacto enorme sobre los requisitos funcionales (y a veces no funcionales) y ya ha visto que los requisitos incompletos y mal especificados pueden ser la razón principal de que los proyectos fracasen. En UML 2, los límites del sistema se conocen como el sujeto, y éste es el término que utilizaremos de ahora en adelante.

El sujeto está definido por quién o qué utiliza el sistema (por ejemplo, los actores) y qué beneficios específicos ofrece el sistema a esos actores (por ejemplo, los casos de uso).

El sujeto se dibuja como un cuadro, etiquetado con el nombre del sistema, con los actores dibujados fuera del límite y los casos de uso dentro. Empezará el modelado de los casos de uso solamente con una idea provisional de dónde se encuentra realmente el sujeto. A medida que encuentre los actores y caso de uso, el sujeto está cada vez más definido.

4.3.2. ¿Qué son actores?

Un actor especifica un rol que cierta entidad externa adopta cuando interactúa con su sistema directamente. Puede representar un rol de usuario, o un rol desempeñado por otro sistema o hardware que toca el límite de su sistema.

En UML 2, los actores también pueden representar otros sujetos, proporcionándole una forma de vincular diferentes modelos de caso de uso.

Para entender los actores, es importante entender el concepto de roles. Piense en un rol como un sombrero que un elemento lleva en un contexto particular. Los elementos pueden tener muchos roles simultáneamente y con el tiempo. Esto significa que un rol puede desempeñarse por muchos elementos diferentes simultáneamente y con el tiempo.

Por ejemplo, si hemos identificado el actor `Cliente` para nuestro sistema, la persona real Jim, Ila, Roland y muchas otras pueden desempeñar ese rol. Esas personas pueden desempeñar también otros roles. Por ejemplo, Roland podría también administrar el sistema (el actor `AdministradorSistema`) al igual que utilizarlo como un `Cliente`. El error más fundamental que los principiantes realizan en el modelado de casos de uso es confundir un rol que desempeña algo en el contexto del sistema con el propio elemento. Siempre pregúntese "¿qué rol desem-

peña el elemento con respecto al sistema?". De esta forma, puede encontrar comportamiento común entre muchos elementos diferentes y de esta forma simplificar su modelo de caso de uso.

Los actores están representados en UML como se muestra en la figura 4.3. Se pueden mostrar como un icono de clase estereotipado `<<actor>>` o como el icono de actor de figura de hombre. Ambas formas de notación de actor son válidas, pero muchos modeladores prefieren utilizar la figura de hombre para representar roles que desempeñarán personas, y la forma de icono de clase para representar roles que desempeñarán otros sistemas.

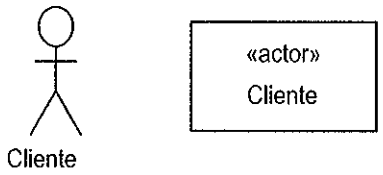


Figura 4.3.

Es importante darse cuenta que los actores son siempre externos al sistema. Por ejemplo, si utiliza un sistema de correo electrónico como una librería online para comprar un libro, entonces usted es externo a ese sistema. Sin embargo, es interesante indicar que, aunque los propios actores son siempre externos al sistema, los sistemas a menudo mantienen alguna representación interna de uno o más actores. Por ejemplo, la librería online mantendrá un objeto de detalles de cliente para la mayoría de clientes que contiene su nombre, dirección y otra información. Ésta es una representación interna del sistema del actor externo `Cliente`. Ahora, es importante ser muy claro sobre esta diferencia, el actor `Cliente` es externo al sistema, pero el sistema podría mantener una clase `DetallesCliente` que es una representación interna de personas que desempeñan el rol del actor `Cliente`.

4.3.2.1. Identificar actores

Para identificar los actores, necesita considerar quién y qué utiliza el sistema y qué roles desempeñan en sus iteraciones con el sistema. Puede llegar a los roles que desempeñan personas y elementos en relación a un sistema al considerar casos de personas y elementos específicos y luego generalizar. Formular las siguientes preguntas le ayudará a identificar actores.

- ¿Quién y qué utiliza el sistema?
- ¿Qué roles desempeñan en la iteración?
- ¿Quién instala el sistema?
- ¿Quién o qué inicia y cierra el sistema?
- ¿Quién mantiene el sistema?
- ¿Qué otros sistemas interactúan con este sistema?

- ¿Quién o qué consigue y proporciona información al sistema?
- ¿Sucede algo en un momento dado?

En términos de modelar actores, recuerde los siguientes puntos:

- Los actores siempre son externos al sistema; están por lo tanto fuera de su control.
- Los actores interactúan directamente con el sistema; así es como ayudan a definir el sujeto.
- Los actores representan roles que personas y elementos desempeñan en relación al sistema, no personas específicas o elementos específicos.
- Una persona o elemento puede desempeñar muchos roles en relación al sistema simultáneamente o con el tiempo. Por ejemplo, si escribiera al igual que impartiera cursos de formación, desde la perspectiva de un sistema de planificación de cursos, desempeñaría dos roles: `Formador` y `AutorCurso`.
- Todo actor necesita un nombre breve que tenga sentido desde la perspectiva del negocio.
- Todo actor debe tener una breve descripción (una o dos líneas) que describa qué es este actor desde una perspectiva de negocio.
- Como las clases, los actores pueden tener compartimentos que muestran atributos del actor y eventos que el actor puede recibir. Normalmente, estos compartimentos no se utilizan demasiado y apenas se muestran en diagramas de caso de uso. Por lo tanto, no los consideraremos.

4.3.2.2. Tiempo como un actor

Cuando necesita modelar elementos que suceden a su sistema en un punto específico en el tiempo pero que no parecen estar activados por ningún actor, puede introducir un actor denominado `Tiempo` según se ilustra en la figura 4.4. Un ejemplo de esto sería un backup automático del sistema que se ejecuta todas las noches.

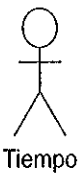


Figura 4.4.

4.3.3. ¿Qué son casos de uso?

The UML Reference Manual [Rumbaugh 1] define un caso de uso como "una especificación de secuencias de acciones, incluidas secuencias variantes y secuencias de

error que un sistema, subsistema o clase puede realizar al interactuar con actores externos".

Un caso de uso es algo que el actor quiere que el sistema haga. Es un "caso de uso" del sistema por un actor específico:

- Los casos de uso se inician siempre por un actor.
- Los casos de uso se escriben siempre desde el punto de vista de los actores.

Normalmente pensamos en los casos de uso en el nivel del sistema, pero como indica la definición, también podemos aplicar casos de uso para describir "casos de uso" de un subsistema (parte de un sistema) o incluso una clase individual. Los casos de uso pueden ser también muy eficaces en el modelado de procesos de negocio, aunque no abordamos ese aspecto en ese libro.

El icono UML para los casos de uso se muestra en la figura 4.5. El nombre del caso de uso se puede escribir dentro o debajo del óvalo.

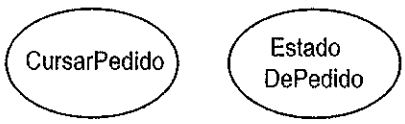


Figura 4.5.

4.3.3.1. Identificar casos de uso

La mejor forma de identificar casos de uso es empezar con la lista de actores y luego considerar cómo cada actor va a utilizar el sistema. Utilizando esta estrategia puede obtener una lista de casos de uso candidatos. Cada caso de uso debe tener asignado un nombre descriptivo, breve, que es una frase verbal; después de todo, el caso de uso es hacer algo. A medida que identifique casos de uso, también puede encontrar algunos nuevos actores. Esto está bien. Algunas veces tiene que considerar la funcionalidad del sistema muy detenidamente antes de encontrar todos los actores o todos los actores correctos.

El modelado de caso de uso es iterativo y continúa vía un proceso de mejora de pasos. Empieza con un nombre para un caso de uso y completa los detalles más adelante. Estos detalles constan de una breve descripción inicial que se convierte en una especificación completa. Aquí tiene una lista de preguntas que puede formular cuando trate de identificar casos de uso.

- ¿Qué funciones querrá un actor específico del sistema?
- ¿El sistema almacena y recupera información? Si es así, ¿qué actores activan este comportamiento?
- ¿Qué sucede cuando el sistema cambia de estado (por ejemplo, iniciar o detener el sistema)? ¿Se notifica a algún actor?
- ¿Afecta algún evento externo al sistema? ¿Qué notifica el sistema sobre estos eventos?

- ¿Interactúa el sistema con algún sistema externo?
- ¿Genera el sistema algún informe?

4.3.3.2. El diagrama de caso de uso

En el diagrama de caso de uso representa el sujeto del modelo de caso de uso por un cuadro etiquetado con el nombre del sujeto. Este cuadro es el sujeto y, como se acaba de mencionar anteriormente, representa el límite del sistema modelado por los casos de uso. Muestra actores fuera del sujeto (externos al sistema) y casos de uso que constituyen el comportamiento del sistema dentro del sujeto, interno al sistema. Esto se ilustra en la figura 4.6.

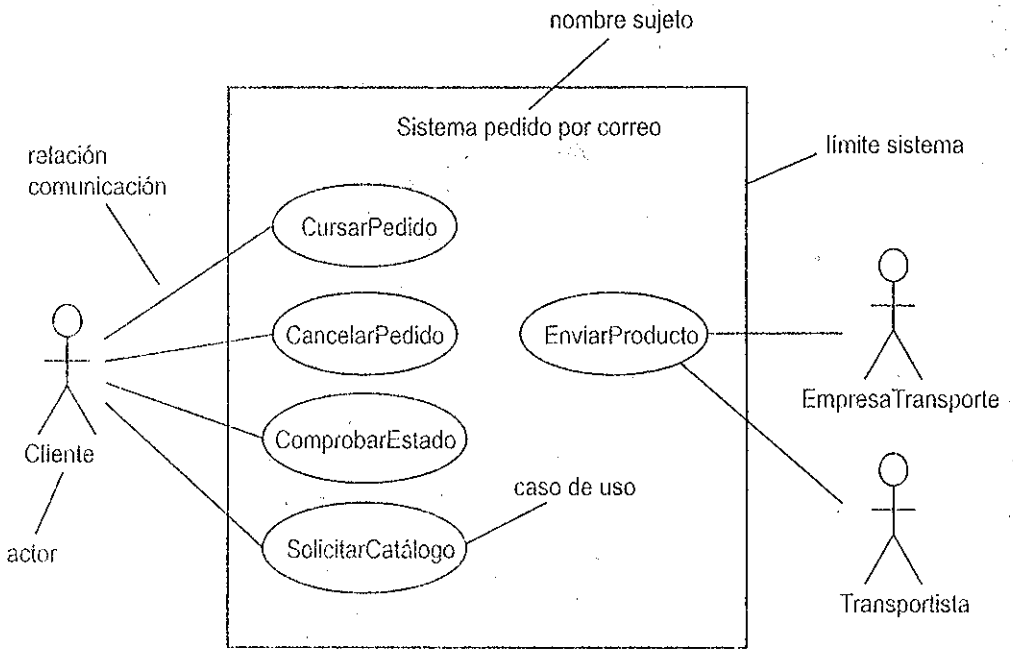


Figura 4.6.

La relación entre un actor y un caso de uso se muestra por una línea continua, que es normalmente el símbolo de asociación de UML. Verá mucho más sobre las asociaciones en el capítulo 9. La asociación entre el actor y el caso de uso indica que el actor y el caso de uso se comunican de alguna forma.

4.3.4. El glosario del proyecto

El glosario del proyecto puede ser uno de los elementos más importantes del proyecto. Todo ámbito de negocio tiene su propio lenguaje único y la finalidad principal de la ingeniería de requisitos y análisis es entender y capturar ese lenguaje. El glosario proporciona un diccionario de términos claves de negocio y definiciones. Debería ser entendible por todo el mundo en el proyecto, incluidos todos los grupos de decisión.

Además de definir los términos clave, el glosario del proyecto debe resolver sinónimos y homónimos.

- Los sinónimos son palabras diferentes que significan lo mismo. Como analista orientado a objetos debe elegir una de estas palabras (la que se utilice más ampliamente) y ajustarse a ella. Las otras variantes se deberían excluir de sus modelos. La razón es que si permite el uso de sinónimos, puede acabar con dos clases que hacen más o menos lo mismo pero tienen nombres diferentes. Igualmente, si permite el uso de todos los sinónimos, puede estar seguro de que la semántica actual de los términos divergirá gradualmente con el tiempo.
- Los homónimos ocurren cuando la misma palabra significa cosas diferentes a diferentes personas. Esto siempre da lugar a problemas de comunicación ya que las diferentes partes hablan diferentes idiomas cuando creen que hablan el mismo idioma. Nuevamente, la forma de resolver esto es elegir un significado para el término y quizás introducir nuevos términos para los otros homónimos.

En el glosario del proyecto, debería grabar el término preferido y listar cualquier sinónimo bajo la definición. Esto puede implicar animar a sus grupos de decisión del negocio a que se acostumbren a diferente terminología. A menudo, es una tarea difícil conseguir que los grupos de decisión cambien su uso del idioma y con mucha persistencia se puede conseguir. UML no establece ningún estándar para un glosario de proyecto. Es una buena práctica mantenerlo lo más sencillo y conciso posible. Utilice un formato como el de un diccionario con una lista ordenada alfabéticamente de palabras y definiciones. Un documento basado en texto puede ser suficiente, pero los proyectos más grandes pueden necesitar un glosario online basado en HTML o XML o incluso una base de datos sencilla. Recuerde que cuanto más accesible y fácil de utilizar sea el glosario, más positivo será el impacto que tendrá sobre el proyecto. Puede ver parte de un glosario de proyecto de ejemplo en la tabla 4.1. Como parte del estilo, siempre escribimos "Ninguno" si no existen sinónimos u homónimos, en lugar de dejar los espacios en blanco u omitirlos. Esto muestra que hemos considerado el término.

Tabla 4.1. Glosario de proyecto para la plataforma de comercio electrónico de Clear View Training.

Término	Definición
Catálogo.	Un listado de todos los productos que en estos momentos se ofrecen a la venta. Sinónimos: Ninguno Homónimos: Ninguno
Pagar.	Una analogía electrónica de pagar en un supermercado en el mundo real. Un lugar donde los clientes pueden pagar los artículos en su carrito de la compra.

Término	Definición
	Sinónimos: Ninguno
	Homónimos: Ninguno
Clear View Training.	Una empresa especializada en venta de libros y CD.
	Sinónimos: CVT
	Homónimos: Ninguno
Tarjeta de crédito.	Una tarjeta como Visa o Mastercard que se puede utilizar para pagar productos.
	Sinónimos: Tarjeta
	Homónimos: Ninguno
Cliente.	Una parte que compra productos o servicios en Clear View Training.
	Sinónimos: Ninguno
	Homónimos: Ninguno

Una consideración con el glosario del proyecto es que los términos y definiciones en el glosario también se utilizarán en el modelo UML. Tiene que asegurase de que los dos documentos se mantienen sincronizados. Desafortunadamente, la mayor parte de las herramientas de modelado UML no proporcionan ningún soporte para esto y normalmente es una actividad manual.

4.4. Actividad UP: Detallar un caso de uso

Habiendo creado un diagrama de caso de uso e identificado los actores y casos de uso claves, necesita empezar a especificar cada caso de uso. Ésta es la actividad de UP conocida como `Detallar un caso de uso`. Se resume en la figura 4.7. Es importante que observe que en este punto no sigue una secuencia exacta y puede elegir especificar algunos o todos los casos de uso a medida que los encuentre. Siempre es difícil presentar actividades paralelas en un libro que, por su propia naturaleza, es lineal. El resultado de esta actividad es un caso de uso más detallado. Esto consta de al menos el nombre del caso de uso y una especificación de caso de uso.

4.5. Especificación de caso de uso

No existe estándar UML para una especificación de caso de uso. Sin embargo, la plantilla mostrada en la figura 4.8 se utiliza comúnmente. Existen plantillas más

complejas, pero en nuestra experiencia es mejor mantener el modelado de caso de uso lo más sencillo posible.

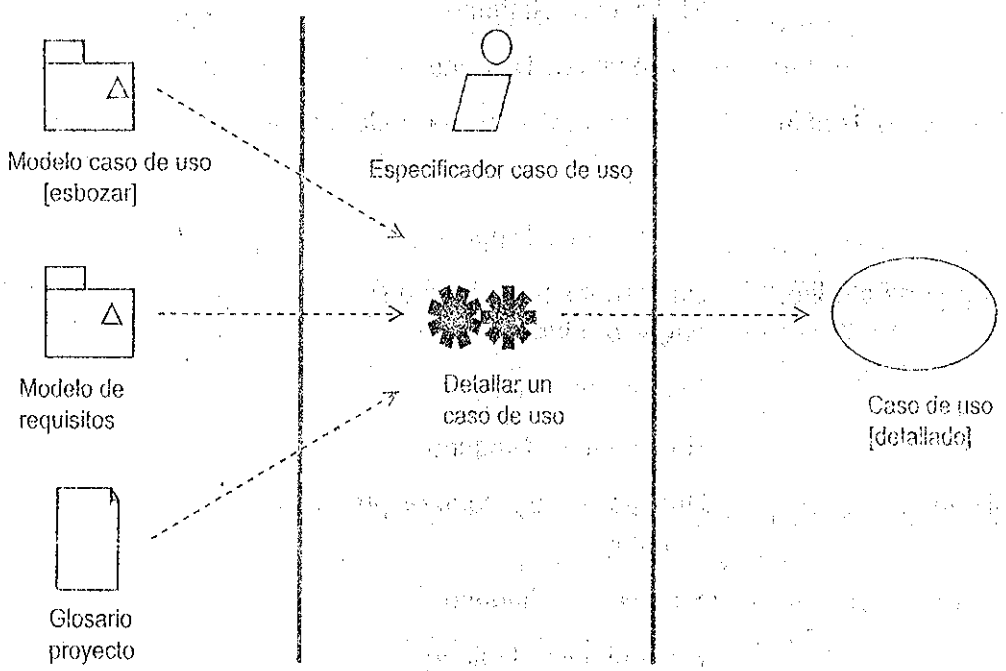


Figura 4.7. Adaptada de la figura 7.14 [Jacobson 1] con permiso de Addison-Wesley.

nombre caso de uso	Caso de uso: PagarImpuestoVentas
identificador caso de uso	ID: 1
descripción breve	Breve descripción: Pagar Impuesto Ventas a Autoridad al final del trimestre.
actores implicados en el caso de uso	Actores principales: Tiempo
	Actores secundarios: Autoridad
el estado del sistema antes de que el caso de uso pueda empezar	Precondiciones: 1. Es el final del trimestre.
los pasos del caso de uso	Flujo principal: actor de tiempo implícito 1. El caso de uso empieza cuando es el final del trimestre. 2. El sistema determina la cantidad de Impuesto Ventas debido a la Autoridad. 3. El sistema envía un pago electrónco a la Autoridad.
el estado del sistema cuando el caso de uso ha terminado	Postcondiciones: 1. La Autoridad recibe la cantidad correcta de Impuesto Ventas.
flujos alternativos	Flujos alternativos: Ninguno.

Figura 4.8.

Es importante que su organización decida un estándar para las especificaciones de caso de uso que se utilice coherentemente dentro de los proyectos. Hemos trabajado en empresas donde no existe tal estándar y puede hacer que todo el proceso de

modelado de caso de uso sea innecesariamente difícil. Existen muchos formatos diferentes, niveles de detalle e incluso interpretaciones de lo que es, o no es un caso de uso incluso dentro del mismo proyecto. Un estándar sencillo y eficaz para las especificaciones de caso de uso puede ayudar a asegurar que su proyecto tiene éxito con el análisis del caso de uso. Presentamos dicho estándar en este capítulo y en el siguiente.

Nuestra plantilla para una especificación sencilla de caso de uso contiene la siguiente información:

- Nombre del caso de uso.
- ID del caso de uso.
- Breve descripción: Un párrafo que captura el objetivo del caso de uso.
- Actores implicados en el caso de uso.
- Precondiciones: Condiciones que deben ser verdaderas antes de que el caso de uso pueda ejecutarse; son restricciones en el estado del sistema.
- Flujo principal: los pasos en el caso de uso.
- Poscondiciones: Condiciones que deben ser ciertas al final del caso de uso.
- Flujos alternativos: Una lista de alternativas al flujo principal.

Cuando examinemos casos de uso más complicados más adelante, los añadiremos a la plantilla para acomodar esa información adicional.

El caso de uso en la figura 4.8 vá sobre el pago del IVA, una forma de impuesto sobre las ventas en muchos países. En este ejemplo, la autoridad correspondiente siempre consigue su impuesto de una forma u otra, y por lo tanto lo establecemos como una poscondición del caso de uso.

Una buena forma de escribir un caso de uso es utilizar "español estructurado" (o alemán, o cualquiera que sea su idioma). En los siguientes apartados presentamos un estilo sencillo que puede utilizar para expresar un caso de uso de forma eficaz.

4.5.1. Nombre del caso de uso

No existe estándar UML para nombrar casos de uso. Siempre nombramos casos de uso poniendo en mayúscula la primera letra de cada palabra con la primera letra de todas en mayúscula. Las palabras del nombre del caso de uso se unen y cada palabra empieza con una letra en mayúscula. Los casos de uso describen el comportamiento del sistema, por lo que el nombre del caso de uso debería ser siempre un verbo o una frase verbal como PagarImpuesto. Siempre debería tratar de elegir un nombre que sea breve, pero descriptivo. Un lector de su modelo de caso de uso debería hacerse una idea clara de la función o proceso de negocio que el caso de uso está modelando solamente por el nombre del caso de uso. Verá numerosos ejemplos de nombres de caso de uso en este capítulo y en el siguiente. El nombre del

caso de uso proporciona un identificador único para el caso de uso dentro de su modelo de caso de uso.

4.5.2. ID de caso de uso

Aunque los nombres del caso de uso deben ser únicos, dentro de su modelo de caso de uso, es posible que cambien con el tiempo. Es posible que desee añadir otro identificador inmutable que identifique de forma única un caso de uso particular dentro de su proyecto. Nosotros a menudo utilizamos un número.

Cuando se trabaja con flujos alternativos, puede elegir utilizar un sistema jerárquico de numeración de modo que el flujo alternativo se pueda vincular con el flujo principal. Por ejemplo, si un caso de uso está numerado X, entonces sus flujos alternativos están numerados X.1, X.2,..., X.n.

4.5.3. Breve descripción

Esto debería ser un solo párrafo que resuma el objetivo del caso de uso. Trate de captar la esencia del caso de uso, el beneficio de negocio que proporciona a sus actores.

4.5.4. Actores

Desde el punto de vista de un caso de uso específico, existen dos tipos de actores:

- **Actores principales:** Estos actores activan el caso de uso.
- **Actores secundarios:** Estos actores interactúan con el caso de uso después de haberse activado.

Cada caso de uso siempre se activa por un solo actor. Sin embargo, el mismo caso de uso puede activarse por diferentes actores en diferentes momentos en el tiempo. Cada actor que puede activar el caso de uso es un actor principal. El resto de actores son actores secundarios.

4.5.5. Precondiciones y poscondiciones

Las precondiciones y poscondiciones son restricciones.

- Las precondiciones restringen el estado del sistema antes de que el caso de uso pueda empezar. Piense en ellas como guardianes que impiden que un actor active el caso de uso hasta que se cumplan todas sus condiciones.
- Las poscondiciones restringen el estado del sistema después de que el caso de uso se ha ejecutado.

Otra forma de examinar esto es que las precondiciones especifican lo que debe ser cierto antes de que el caso de uso se pueda activar, y las poscondiciones especi-

fican qué será verdadero después de que el caso de uso se haya ejecutado. Las precondiciones y poscondiciones le ayudan a diseñar sistemas que funcionan correctamente.

Las precondiciones y poscondiciones siempre deberían ser declaraciones sencillas sobre el estado del sistema que evaluará como verdadero y falso; esto se conoce como condiciones booleanas.

Si su caso de uso no tiene ninguna precondición o poscondición, entonces se debería escribir "Ninguno" en el apartado apropiado de la especificación del caso de uso. Esto demuestra que ha considerado el asunto, mientras que dejar el apartado en blanco es ambiguo.

4.5.6. Flujo principal

Los pasos en un caso de uso se listan en un flujo de eventos. Puede pensar en un caso de uso como el delta de un río con muchos canales. Cada caso de uso tiene un flujo principal que es el canal principal hacia el delta. Los otros canales más pequeños en el delta son flujos alternativos. Estos flujos alternativos pueden capturar errores e interrumpir el flujo principal. El flujo principal a veces se conoce como el escenario principal, y los flujos alternativos como escenarios secundarios.

El flujo principal lista los pasos en un caso de uso que capturan el "mundo perfecto", situación donde todo sucede según lo esperado y deseado, y no existen errores, desviaciones o interrupciones.

Puede modelar las desviaciones al flujo principal de dos formas, que se tratarán en breve.

1. Desviaciones simples: Crea ramificaciones en el flujo principal.
2. Desviaciones complejas: Escribe flujos alternativos.

El flujo principal siempre empieza por el actor principal haciendo algo para activar el caso de uso. Una buena forma de iniciar un flujo de eventos es como sigue:

```
1. El caso de uso empieza cuando un <actor><función>
```

Recuerde que el tiempo puede ser un actor, por lo tanto el caso de uso puede también empezar con una expresión de tiempo en lugar del actor, como muestra la figura 4.8.

El flujo de eventos consta de una secuencia de pasos breves que son declarativos, están numerados y ordenados en el tiempo. Cada paso en el flujo de caso de uso debería estar en la forma de

```
<número>El<algo><alguna acción>
```

El flujo de caso de uso de los eventos también se puede capturar como prosa. Sin embargo, no lo recomendamos ya que es demasiado impreciso. Aquí tiene un ejemplo de un par de pasos en un caso de uso `CursarPedido`.

```
1. El caso de uso empieza cuando el cliente selecciona "cursar pedido".
2. El cliente escribe su nombre y dirección en el formulario.
```

Éstos son casos correctos. En ambos casos tenemos una sencilla declaración declarativa de algo realizando alguna acción. Un ejemplo de un paso de caso de uso mal formado sería como el siguiente:

```
2. Se incorporan los detalles del cliente.
```

De hecho, cualquier paso escrito en pasiva está mal formado. Este caso en particular contiene tres eliminaciones importantes:

- ¿Quién incorpora los detalles del cliente?
- ¿En qué se incorporan los detalles del cliente?
- ¿Qué son específicamente "detalles del cliente"?

Es importante que reconozca y evite eliminaciones cuando escriba flujos de caso de uso. Aunque es posible que lo pueda saber por el contexto o averiguar lo que significa, éste no es el caso. El caso es que el caso de uso debería ser una declaración precisa de una pieza de la funcionalidad del sistema.

Cuando encuentra vaguedad, eliminaciones o generalizaciones durante el proceso de análisis, es de utilidad hacerse las siguientes preguntas:

- ¿Quién específicamente...?
- ¿Qué específicamente...?
- ¿Cuándo específicamente...?
- ¿Dónde específicamente...?

4.5.6.1. Ramificación dentro de un flujo

La especificación UML no especifica ninguna forma de mostrar ramificaciones dentro de un flujo. Utilizamos un modismo para mostrar la ramificación de una forma sencilla sin tener que escribir un flujo alternativo aparte. Utilizamos la palabra clave `Si` para indicar una ramificación.

Merece la pena saber que algunos modeladores de caso de uso pueden desaprobar la ramificación dentro de casos de uso. Argumentan que siempre que hay alguna ramificación, se debería escribir un nuevo flujo alternativo. En realidad, este argumento merece la pena tenerlo en cuenta; sin embargo, tomaremos el enfoque más pragmático de que una pequeña cantidad de ramificación en un flujo es deseable porque reduce el número total de flujos alternativos y conduce a una representación más compacta de los requisitos.

4.5.6.2. Palabra clave Si

Utilice la palabra clave `Si` para indicar una ramificación en un flujo. El ejemplo en la figura 4.9 muestra un flujo de eventos bien estructurado con dos ramificacio-

nes. Toda ramificación va prefijada con la palabra clave `Si` y empieza con una sencilla expresión booleana como `Si el usuario escribe una nueva cantidad`, lo que es verdadero o falso. El texto sangrado bajo la declaración `Si` es lo que sucederá si la expresión booleana es verdadera. Puede indicar claramente el cuerpo de la declaración `Si` por medio del uso de la sangría y la numeración sin necesidad de introducir alguna otra sintaxis de cierre de declaración.

Caso de uso: GestionarCarro
ID: 2
Breve descripción: El Cliente cambia la cantidad de un artículo en el carro.
Actores principales: Cliente
Actores secundarios: Ninguno
Precondiciones: 1. Los contenidos del carro son visibles.
Flujo principal: 1. El caso de uso empieza cuando el Cliente selecciona un artículo en el carro. 2. Si el Cliente selecciona "eliminar artículo" 2.1 El sistema elimina el artículo del carro. 3. Si el Cliente escribe una nueva cantidad 3.1 El sistema actualiza la cantidad del artículo en el carro.
Postcondiciones Ninguna.
Flujos alternativos: Ninguno

Figura 4.9.

La ramificación puede reducir el número de poscondiciones de caso de uso. Esto es porque los pasos dentro de una ramificación pueden o no ocurrir, dependiendo de las circunstancias. No pueden generar poscondiciones, que son hechos que deben ser verdaderos en lugar de hechos que pueden ser verdaderos.

4.5.6.3. Repetición dentro de un flujo

Algunas veces tiene que repetir una acción varias veces dentro de un flujo de eventos. Esto no ocurre demasiado a menudo en el modelado de casos de uso, pero cuando lo hace, es de utilidad contar con una estrategia.

La especificación UML no especifica ninguna forma de mostrar repetición dentro de un flujo, por lo que incorporamos sencillas palabras clave `Para` y `Mientras`.

4.5.6.4. Palabra clave Para

Puede modelar la repetición al utilizar la palabra clave `Para`. El formato es el siguiente:

```
n. Para (expresión de iteración)
   n.1. Hacer algo
   n.2. Hacer algo más
   n.3. ...
n+1.
```

La expresión de iteración es cierta expresión que evalúa en positivo un número completo de iteraciones. Cada línea sangrada detrás de la declaración `Para` se repite para el número de iteraciones especificado en la expresión de la iteración. Un ejemplo se facilita en la figura 4.10.

Caso de uso: EncontrarProducto
ID: 3
Breve descripción: El sistema encuentra productos basándose en los criterios de búsqueda del Cliente y se los muestra.
Actores principales: Cliente
Actores secundarios: Ninguno
Precondiciones: Ninguna
Flujo principal: 1. El caso de uso empieza cuando el Cliente selecciona "encontrar producto" 2. El sistema pregunta al Cliente los criterios de búsqueda. 3. El Cliente escribe los criterios solicitados. 4. El sistema busca productos que coincidan con criterios del Cliente. 5. Si el sistema encuentra algún producto que coincide entonces 5.1 Para cada producto encontrado 5.1.1 El sistema muestra una imagen del producto. 5.1.2 El sistema muestra un resumen de los detalles del producto. 5.1.3 El sistema muestra el precio del producto. 6. Sino 6.1 El sistema le dice al Cliente que no se han encontrado productos.
Postcondiciones: Ninguna.
Flujos alternativos: Ninguno.

Figura 4.10.

4.5.6.5. Palabra clave Mientras

Utilice la palabra clave `Mientras` para modelar una secuencia de acciones en el flujo de eventos que se realiza mientras cierta condición booleana es verdadera. El formato es el siguiente:

```
n. Mientras (condición booleana)
   n.1. Hacer algo
   n.2. Hacer algo más
   n.3. ...
n+1.
```

Igual que `Para`, la palabra clave `Mientras` se utiliza poco a menudo. Un ejemplo se muestra en la figura 4.11. La secuencia de líneas sangradas detrás de la declaración `Mientras` se repite hasta que la condición booleana especificada en la cláusula `Mientras` se hace falsa.

Caso de uso: MostrarDetallesEmpresa
ID: 4
Breve descripción: El sistema muestra los detalles de la empresa al Cliente.
Actores principales: Cliente
Actores secundarios: Ninguno.
Precondiciones: Ninguna.
Flujo principal: 1. El caso de uso empieza cuando Cliente selecciona "mostrar detalles empresa" 2. El sistema muestra una página Web con los detalles de la empresa. 3. Mientras el Cliente examina los detalles de la empresa 3.1 El sistema hace sonar música de fondo. 3.2 El sistema muestra ofertas especiales en un banner.
Postcondiciones: 1. El sistema ha mostrado los detalles de la empresa. 2. El sistema ha hecho sonar música de fondo. 3. El sistema ha mostrado ofertas especiales.
Flujos alternativos: Ninguno.

Figura 4.11.

4.5.7. Modelar flujos alternativos

Todo caso de uso tiene un flujo principal y puede tener muchos flujos alternativos. Éstos son rutas de acceso alternativas a través del caso de uso que capturan errores, ramificaciones e interrupciones en el flujo principal. Como acaba de ver, la especificación del caso de uso contiene el flujo principal y una lista de los nombres de los flujos alternativos.

El punto clave sobre los flujos alternativos es que frecuentemente no regresan al flujo principal. Esto es porque los flujos alternativos a menudo tratan con errores y excepciones al flujo principal y tienden a tener diferentes poscondiciones. Puede ver los flujos alternativos ilustrados en la figura 4.12.

Puede documentar los flujos alternativos de forma separada o anexarlos al final del caso de uso. Nosotros preferimos documentarlos aparte.

Como ejemplo de cómo puede modelar un caso de uso con flujos alternativos, considere la figura 4.13.

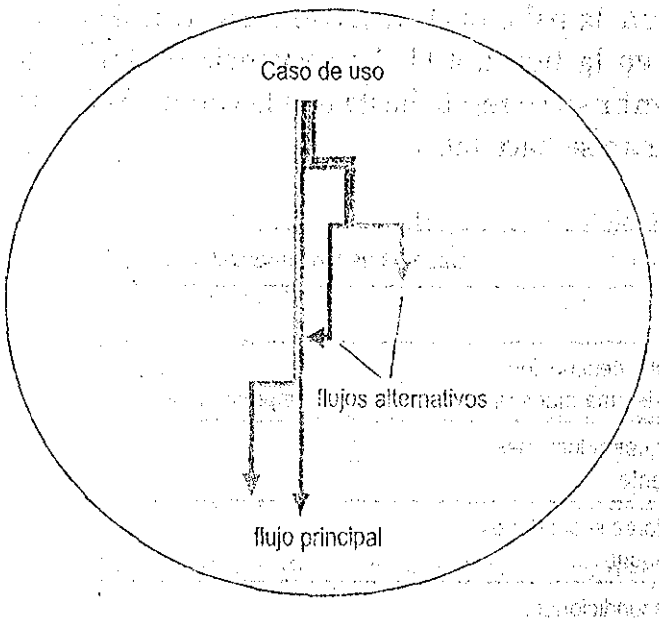


Figura 4.12.

Caso de uso: CrearNuevaCuentaCliente
ID: 5
Breve descripción: El sistema crea una nueva cuenta para el Cliente.
Actores principales: Cliente
Actores secundarios: Ninguno.
Precondiciones: Ninguna.
Flujo principal: 1. El caso de uso empieza cuando el Cliente selecciona "crear nueva cuenta cliente". 2. Mientras que los detalles del Cliente son no válidos 2.1 El sistema pide al Cliente que escriba sus detalles de correo electrónico, contraseña, y contraseña de nuevo para confirmación. 2.2 El sistema valida los detalles del Cliente. 3. El sistema crea una nueva cuenta para el Cliente.
Postcondiciones: 1. Se ha creado una nueva cuenta para el Cliente.
Flujos alternativos: DirecciónCorreoInválida ContraseñaInválida Cancelar

Figura 4.13.

Puede ver que este caso de uso tiene tres flujos alternativos, `DirecciónCorreoInválida`, `ContraseñaInválida`, y `Cancelar`. En la figura 4.14 hemos documentado el flujo alternativo `DirecciónCorreoInválida`.

Flujo alternativo: CrearNuevaCuentaCliente: DirecciónCorreoInválida
ID: 5.1
Breve descripción: El sistema informa al Cliente que ha facilitado una dirección de correo inválida.
Actores principales: Cliente
Actores secundarios: Ninguno.
Precondiciones: 1. El Cliente ha facilitado una dirección de correo inválida.
Flujo alternativo: 1. El flujo alternativo empieza después del paso 2.2 del flujo principal. 2. El sistema informa al Cliente que ha facilitado una dirección de correo inválida.
Postcondiciones: Ninguna.

Figura 4.14.

Observe que podemos realizar varios cambios en la plantilla del caso de uso para acomodar los flujos alternativos.

- `Nombre`: Utilizamos la siguiente convención de nombre para flujos alternativos:

```
Flujo alternativo: CrearNuevaCuentaCliente:DirecciónCorreoInválida
```

  Esto indica que es un flujo alternativo denominado `DirecciónCorreoInválida` para el caso de uso `CrearNuevaCuentaCliente`.

- `ID`: Observe cómo hemos utilizado un sistema de numeración jerárquico para relacionar el flujo alternativo con el caso de uso principal.
- `Actores`: Lista los actores utilizados por el flujo alternativo.
- `Precondiciones` y `poscondiciones`: Los flujos alternativos pueden tener su propio conjunto de precondiciones y poscondiciones que son diferentes de los del caso de uso. Si el flujo alternativo regresa al flujo principal, entonces sus poscondiciones se añaden eficazmente a las del flujo principal.
- `Flujo alternativo`: Los pasos en flujo alternativo.
- Un flujo alternativo no debería tener flujos alternativos: De lo contrario, se hacen demasiado complejos muy rápidamente.

Los flujos alternativos se pueden activar de tres formas diferentes:

1. El flujo alternativo se puede activar en lugar del flujo principal.
2. El flujo alternativo se puede activar después de un paso en particular en el flujo principal.

3. El flujo alternativo se puede activar en cualquier momento durante el flujo principal.

Cuando un flujo alternativo se ejecuta en lugar del flujo principal, se activa por el actor principal y reemplaza a todo el caso de uso.

Cuando el flujo alternativo se activa después de un paso en particular en el flujo principal, debería empezar de la siguiente forma:

```
1. El flujo alternativo empieza detrás del paso X del flujo principal.
```

Ésta es una forma de ramificación, pero es diferente de la ramificación que hemos tratado anteriormente porque es una desviación importante del flujo principal y podría no regresar a él.

Cuando un flujo alternativo se puede activar en cualquier momento durante el flujo principal, debería empezar de la siguiente forma:

```
1. El flujo alternativo empieza en cualquier momento.
```

Utilice este tipo de flujo alternativo para modelar algo que puede ocurrir en cualquier momento en el flujo principal antes del último paso. Por ejemplo, en el caso de uso `CrearNuevaCuentaCliente`, el `Cliente` puede elegir Cancelar la creación de la cuenta en cualquier momento. Puede documentar `Cancelar` como se muestra en la figura 4.15.

Flujo alternativo: CrearNuevaCuentaCliente:Cancelar
ID: 5.2
Breve descripción: El Cliente cancela el proceso de creación de cuenta.
Actores principales: Cliente
Actores secundarios: Ninguno.
Precondiciones: Ninguna.
Flujo alternativo: 1. El flujo alternativo empieza en cualquier momento. 2. El Cliente cancela la creación de cuenta.
Postcondiciones: 1. No se ha creado una nueva cuenta para el Cliente.

Figura 4.15.

Si quisiera que el flujo alternativo regresara al flujo principal, podría expresarlo de la siguiente manera:

```
N. El flujo alternativo regresa al paso M del flujo principal.
```

En este ejemplo, el flujo alternativo ejecuta su último paso N, y luego la ejecución del flujo principal continúa en el paso M.

4.5.7.1. Encontrar flujos alternativos

Puede identificar flujos alternativos al inspeccionar el flujo principal. En cada caso del flujo principal busque:

- Alternativas posibles al flujo principal.
- Errores que puedan surgir en el flujo principal.
- Interrupciones que puedan ocurrir en un punto particular en el flujo principal.
- Interrupciones que puedan ocurrir en cualquier punto en el flujo principal.

Cada una de éstas es una fuente posible de un flujo alternativo.

4.5.7.2. ¿Cuántos flujos alternativos?

Como hemos dicho, existe exactamente un flujo principal por caso de uso. Sin embargo, existen muchos flujos alternativos. La pregunta es ¿cuántos? Debería tratar de limitar el número de flujos alternativos al mínimo necesario. Existen dos estrategias para esto.

- Seleccione los flujos alternativos más importantes y documente estos.
- Donde existen grupos de flujos alternativos que son todos muy similares, documente un miembro del grupo como ejemplo y (si fuera necesario) añada notas a esto explicando cómo difieren de éste.

Retomando la analogía del delta del río, además del canal principal, puede haber muchas ramificaciones y flujos alternativos en el delta. No puede permitirse mapear todos ellos, por lo que elige solamente los principales. Igualmente, muchas de estas ramificaciones fluyen en casi la misma dirección con sólo pequeñas diferencias. Por lo tanto, puede mapear un canal en detalle y proporcionar notas explicando cómo los otros canales más pequeños se desvían de éste. Ésta es una forma eficaz y eficiente de modelar un caso de uso complejo.

El principio básico en el modelado de caso de uso es mantener la cantidad de información capturada en el mínimo necesario. Esto significa que muchos flujos alternativos nunca se pueden especificar del todo; una descripción de una línea añadida al caso de uso puede ser suficiente detalle para permitir entender el funcionamiento del sistema. Éste es un punto importante. Es fácil verse inundado por flujos alternativos, y hemos visto más de una actividad de modelado de caso de uso fracasar por esto.

Recuerde que captura casos de uso para entender el comportamiento deseado del sistema y no por el hecho de crear un modelo completo de caso de uso. Por lo tanto, deja de modelar casos de uso cuando cree que ha conseguido ese entendimiento. También, puesto que el UP es un ciclo de vida iterativo, siempre puede regresar a un caso de uso y realizar más trabajo si hay algún aspecto del comportamiento del sistema que decide que no entiende.

4.6. Seguimiento de requisitos

Con un modelo de requisitos y un modelo de caso de uso, tiene dos "bases de datos" de requisitos funcionales. Es importante relacionar las dos para averiguar si hay algo en su modelo de requisitos que no está tratado por el modelo de caso de uso y viceversa. Éste es un aspecto del seguimiento de requisitos.

Seguir requisitos funcionales de casos de uso es complicado por el hecho de que existe una relación muchos a muchos entre requisitos funcionales individuales y casos de uso. Un caso de uso tratará muchos requisitos funcionales individuales y un requisito funcional puede estar presente en varios casos de uso diferentes.

Afortunadamente, tendrá soporte de herramientas de modelado para el seguimiento de requisitos y, de hecho, herramientas de ingeniería de requisitos como RequisitePro y DOORS le permiten vincular requisitos individuales en su base de datos de requisitos con casos de uso específicos y viceversa. De hecho, UML proporciona un soporte bastante bueno para el seguimiento de requisitos. Utilizando valores etiquetados, puede asociar una lista de números ID de requisito con cada caso de uso. En la herramienta de requisitos, puede vincular uno o más identificadores de caso de uso con requisitos específicos.

Si no tiene ningún soporte de herramienta de modelado, tiene que hacer el trabajo manualmente. Un buen enfoque es crear una matriz de trazabilidad de requisitos. Esto es una sencilla cuadrícula con los números ID de los requisitos individuales hacia abajo en un eje y los nombres de caso de uso (y/o números ID) a lo largo del otro. Una cruz se sitúa en todas las celdas donde se cruzan un caso de uso y requisito. Las matrices de trazabilidad de requisitos se crean a menudo en hojas de cálculo. Un ejemplo se proporciona en la tabla 4.2.

Tabla 4.2.

		Caso de uso			
		CU1	CU2	CU3	CU4
Requisito	R1	X			
	R2	X	X		
	R3			X	
	R4				X
	R5	X			

Una matriz de trazabilidad de requisitos es una herramienta de utilidad para comprobar coherencias. Si existe un requisito que no mapea con ningún caso de uso, entonces falta un caso de uso. Y al contrario, si hay un caso de uso que no mapea con ningún requisito, sabe que su conjunto de requisitos está incompleto.

Con el conjunto de herramientas SUMR que hemos tratado anteriormente, puede automatizar la creación de una posible matriz de trazabilidad de requisitos. La idea es sencilla: si una palabra en el glosario del proyecto ocurre en un requisito y en un caso de uso, existe una probabilidad mayor que las dos estén relacionadas de alguna forma. Esto crea una posible matriz de trazabilidad de requisitos. Lo denominamos "posible", porque dicho análisis textual tendrá errores y omisiones y la matriz necesita inspeccionarse manualmente. Aún así puede ahorrar mucho tiempo y puede ayudar a los ingenieros de requisitos a realizar una tarea laboriosa que de otra forma podría no hacerse.

4.7. ¿Cuándo aplicar modelado de caso de uso?

Los casos de uso capturan requisitos funcionales y por lo tanto no son eficaces para sistemas dominados por requisitos no funcionales.

Los casos de uso son la mejor opción para la captura de requisitos cuando:

- El sistema está dominado por requisitos funcionales.
- El sistema tiene muchos tipos de usuarios para el que presenta diferente funcionalidad (existen muchos actores).
- El sistema tiene muchas interfaces (existen muchos actores).

Los casos de uso no serán una buena opción cuando:

- El sistema está dominado por requisitos no funcionales.
- El sistema tiene pocos usuarios.
- El sistema tiene pocas interfaces.

Ejemplos de sistemas donde los casos de uso pueden no ser apropiados son sistemas incorporados y sistemas que son algorítmicamente complejos pero con pocos interfaces. Para estos sistemas, sería mucho mejor que optara por técnicas de ingeniería de requisitos más convencionales. Es una cuestión de elegir la herramienta correcta para el trabajo entre manos.

4.8. ¿Qué hemos aprendido?

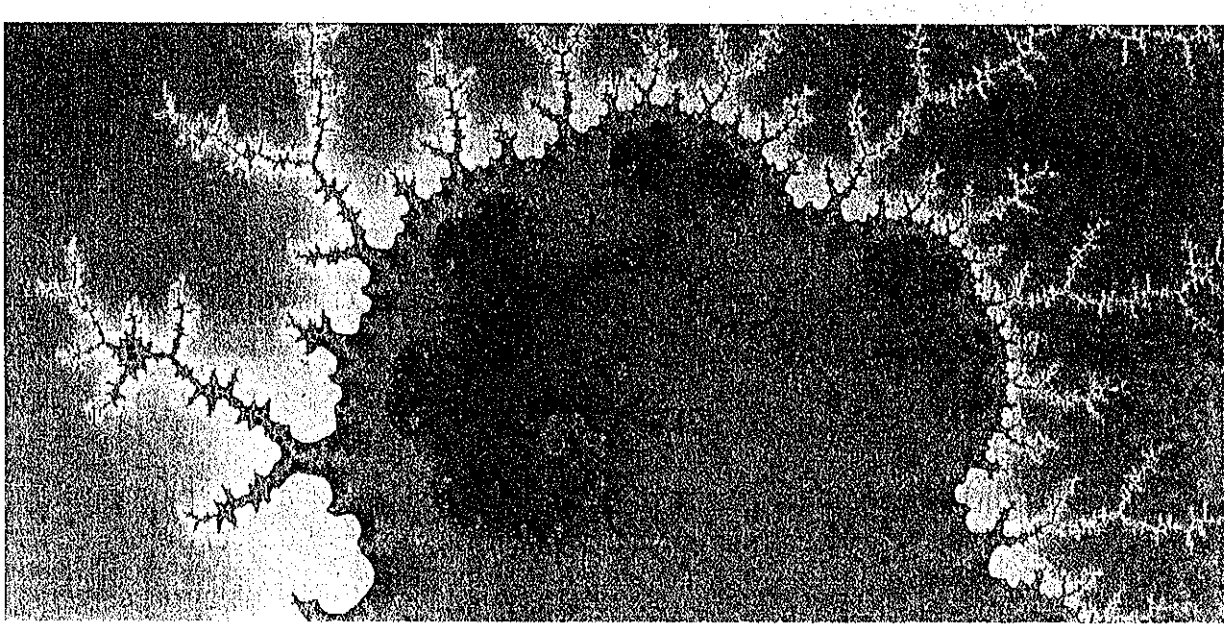
Este capítulo ha tratado sobre capturar requisitos del sistema por el modelado de caso de uso. Ha aprendido lo siguiente.

- La actividad de modelado de caso de uso es parte del workflow de requisitos.
- La mayoría del trabajo en el workflow de requisitos ocurre en las fases de comienzo y elaboración del ciclo de vida del proyecto UP.

- Las actividades UP claves son `Encontrar actores y casos de uso` y `Detallar un caso de uso`.
- El modelado de caso de uso es otra forma de ingeniería de requisitos que procede de la siguiente forma:
  - Encontrar el sujeto.
  - Encontrar actores.
  - Encontrar casos de uso.
- El sujeto define qué es parte del sistema y qué es externo al sistema.
- Los actores son roles desempeñados por elementos externos al sistema que interactúan directamente con el sistema.
  - Puede encontrar actores al considerar quién o qué utiliza o interactúa directamente con el sistema.
  - El tiempo a veces es un actor.
- Los casos de uso son funciones que el sistema realiza en nombre de actores específicos. Puede encontrar casos de uso al considerar cómo cada actor interactúa con el sistema.
  - Puede encontrar casos de uso al considerar qué funciones ofrece el sistema a los actores.
  - Los casos de uso se inician siempre por el actor.
  - Los casos de uso se escriben siempre desde el punto de vista de los actores.
- El diagrama de caso de uso muestra:
  - El sujeto.
  - Actores.
  - Casos de uso.
  - Interacciones.
- El glosario del proyecto proporciona definiciones de términos clave de negocio; resuelve sinónimos y homónimos.
- La especificación de caso de uso incluye:
  - Un nombre de caso de uso.
  - Un identificador único.
  - Una breve descripción; el objetivo del caso de uso.
  - Actores: Actores primarios, activan el caso de uso, y actores secundarios, interactúan con el caso de uso después de haberlo activado.
  - Precondiciones; restricciones del sistema que afectan a la ejecución de un caso de uso.

- Flujo principal, la secuencia de pasos declarativos, ordenados en el tiempo en el caso de uso.
- Poscondiciones, restricciones del sistema que surgen de la ejecución de un caso de uso.
- Flujos alternativos, una lista de alternativas al flujo principal.

- Puede reducir el número de casos de uso al permitir una cantidad limitada de ramificaciones dentro del flujo de eventos:
  - Utilice la palabra clave `Si` para ramificaciones que ocurren en un paso particular en el flujo.
  - Utilice secciones de `Flujo alternativo` en el caso de uso para capturar ramificaciones que pueden ocurrir en cualquier punto en el flujo.
- Puede mostrar repetición dentro de un flujo al utilizar las palabras clave:
  - `Para (expresión de iteración).`
  - `Mientras (condición booleana).`
- Todo caso de uso tiene un flujo principal; éste es el escenario perfecto donde todo sucede según lo planificado.
- Los casos de uso más complejos pueden tener uno o más flujos alternativos. Éstos son rutas de acceso a través de un caso de uso que representan excepciones, ramificaciones e interrupciones.
- Encuentra flujos alternativos clave al examinar el flujo principal y buscar:
  - Alternativas.
  - Situaciones de error.
  - Interrupciones.
- Solamente descomponga un caso de uso en flujos alternativos cuando añada valor al modelo.
- Los requisitos en el modelo de requisitos se pueden trazar a los casos de uso utilizando una matriz de trazabilidad de requisitos.
- El modelado de casos de uso es más apropiado para sistemas que:
  - Están dominados por requisitos funcionales.
  - Tienen muchos tipos de usuarios.
  - Tienen muchas interfaces a otros sistemas.
- El modelado de casos de uso es menos apropiado para sistemas que:
  - Están dominados por requisitos no funcionales.
  - Tienen pocos usuarios.
  - Tienen pocas interfaces.



5

Modelado avanzado de caso de uso