INSTRUCTIVO PARA EL MODELADO CON CASOS DE USO

Auxiliar: Andrés Neyem

1 Introducción

1.1 Objetivo

El objetivo de este documento es el de establecer pautas para la especificación de requerimientos mediante la técnica de modelado con casos de uso.

1.2 Alcance

Estas pautas abarcan temas relacionados a documentación de casos de uso, paquetes y actores. Estas pautas sirven, a modo de ejemplo, como posible marco general para ser tenido en cuenta por los alumnos.

2 Modelado de paquetes

Un paquete es un elemento de modelado que se emplea para organizar al resto de los elementos de modelado en agrupamientos lógicos. Los paquetes permiten subdividir los modelos en partes manejables según los criterios que establezca el analista, haciéndolos más sencillos de comprender, de manipular y de comunicar. Los casos de uso, al igual que muchos otros elementos de modelado del UML, pueden organizarse en paquetes. Además, los paquetes ayudan a identificar en forma temprana los subsistemas o módulos de un sistema a construir y a organizar al equipo de trabajo.

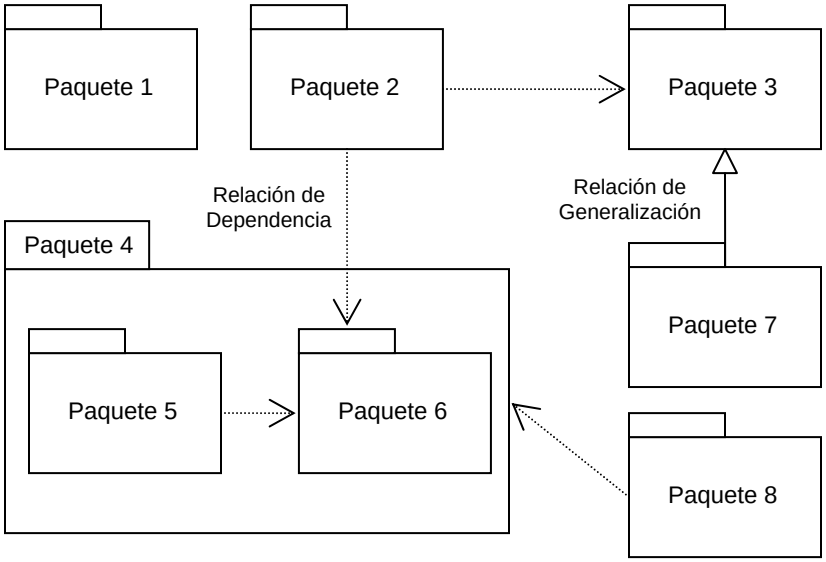
Los paquetes permiten agrupar y aislar casos de uso, clases, asociaciones y eventualmente otros paquetes. Un paquete agrupa normalmente a una serie de entidades que corresponden a una funcionalidad bien definida. A menudo, esta funcionalidad es lo que definirá el nombre del paquete. Los paquetes pueden contener a otros paquetes para posibilitar la descomposición de los modelos jerárquicamente. En la práctica es mejor evitar paquetes muy anidados, siendo lo apropiado aproximadamente tres o cuatro niveles de anidamiento, aunque esto depende de la complejidad del problema.

Cada paquete tiene un nombre que lo distingue de otros paquetes. El nombre debe representar la agrupación lógica de los elementos que contiene el paquete. Cada paquete representa un espacio lógico y los nombres deben ser únicos dentro de cada espacio lógico. Por lo tanto, un nombre de paquete (o de cualquier elemento) debe ser único dentro del paquete contenedor, aunque pueden existir elementos con los mismos nombres en diferentes paquetes, dado que se entiende por nombre completo al nombre del elemento precedido por toda la jerarquía de paquetes que lo contiene.

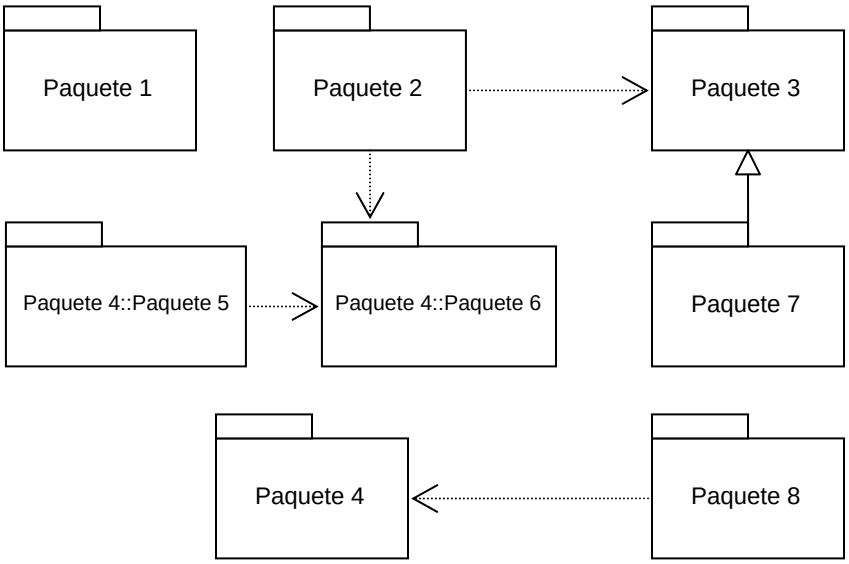
Los paquetes pueden tener relaciones de dependencias que con otros paquetes, de la siguiente manera: a) Generalización, corresponde a generalización entre paquetes, y b) Dependencia, sólo indica las relaciones de dependencia con los paquetes en los que existan elementos que son utilizados en éste.

2.1 Representación Gráfica de Paquetes

2.1.1 Ejemplo A.1



2.1.2 Ejemplo A.2

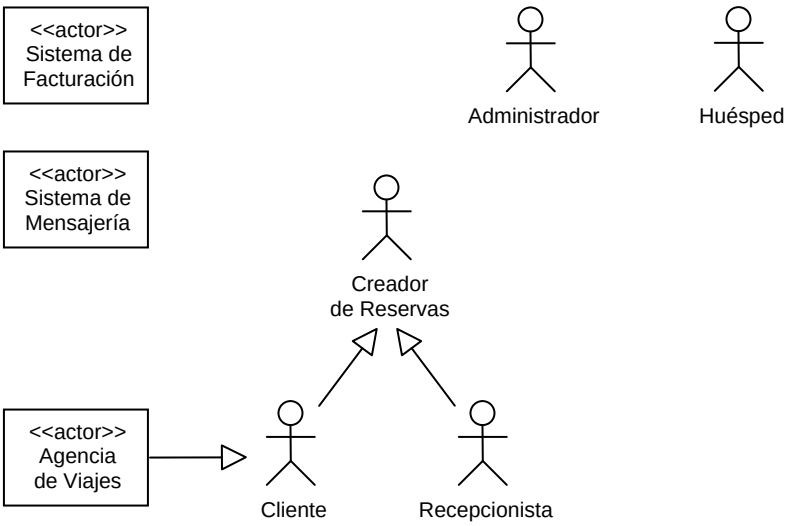


3 Actores

Un actor representa un conjunto coherente de roles que los usuarios de los casos de uso juegan al interactuar con el sistema. Normalmente, un actor representa un rol que es jugado por una persona, un dispositivo de hardware, una base de datos o incluso otro sistema que interactúe.

Un actor puede poseer atributos, que es una propiedad identificada con un nombre, que describe un rango de valores que pueden tomar las instancias de la propiedad. Un atributo es la abstracción de una característica simple del elemento que se está modelando. Por ejemplo, el actor Cliente podría tener los siguientes atributos: nombre, apellido, calle, númeroDeCalle, teléfono, fechaDeNacimiento, etc. La identificación de estos atributos nos ayudará a realizar un análisis más profundo, y nos permitirá en una etapa temprana la identificación de posibles Clases. Cabe aclarar también que no siempre vamos a encontrar atributos en los actores, aunque siempre deberán tener al menos una responsabilidad, caso contrario, no tiene sentido su existencia en el sistema.

3.1 Representación Gráfica de Actores



4 Casos de Uso

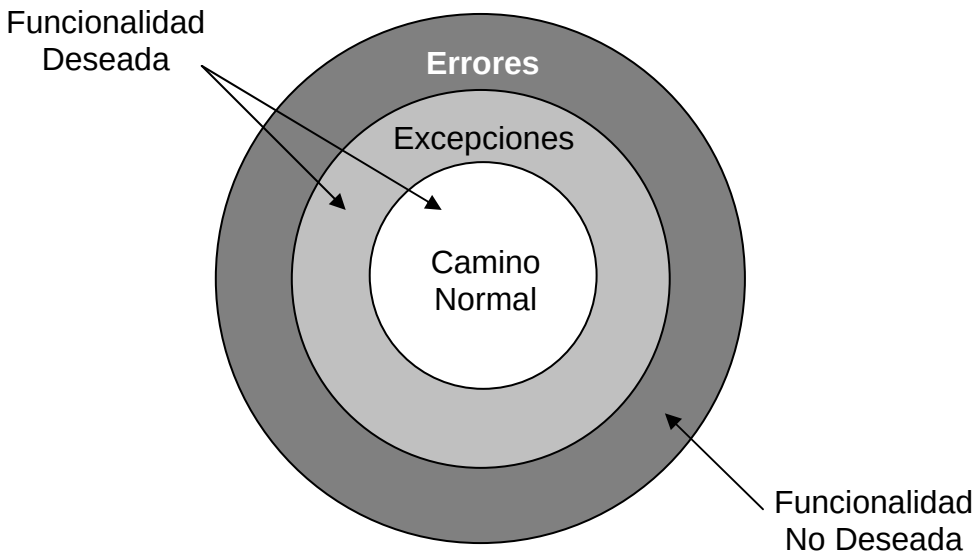
Un caso de uso es la representación abstracta de una funcionalidad del sistema que provee un resultado de valor desde el punto de vista de sus actores. Además de poseer una representación gráfica como todos los elementos del UML, se describe por medio de un conjunto de secuencias de acciones, incluyendo variantes, que ejecuta un sistema para producir un resultado observable, de valor para un actor como el cálculo de un resultado, la generación de un nuevo objeto o el cambio de estado de un objeto.

Una posible plantilla establecida para la descripción de los casos de uso podría constar de tres partes. La primera brinda información general, la segunda informa los actores

involucrado en el caso de uso y la tercera describe su comportamiento, por medio de las secuencias de su curso normal, los subflujos del curso normal y sus excepciones, y aporta consideraciones de diseño.

4.1 Descripción del Comportamiento del Caso de uso

El comportamiento de un caso de uso se puede especificar describiendo un flujo de eventos de forma textual, lo suficientemente claro para que alguien ajeno al sistema lo entienda fácilmente. Cuando se escribe este flujo o curso de eventos se debe incluir el cómo y cuándo se empieza y acaba el caso de uso, cuándo interactúa con los actores y qué objetos se intercambian, el flujo básico o curso normal y los flujos alternativos (subflujos o excepciones) del comportamiento.



4.2 Curso Normal

El curso normal se describe en una sección en la que se deben indicar qué precondiciones (cómo y cuándo comienza el caso de uso) hay que cumplir para que se pueda iniciar el caso de uso. A continuación se explican los pasos que deben llevarse a cabo para la ejecución del caso de uso (flujo de eventos).

Precondición

Se expresa de manera tal que no exista ningún tipo de ambigüedad y que permita objetivamente su comprobación. Las condiciones deben ser encerradas entre corchetes, y se unen mediante algún operador lógico en caso de que existiera más de una condición.
Una expresión lógica es una expresión que al ser evaluada puede dar como resultado, únicamente, verdadero o falso. La sintaxis que deberá emplearse consistirá en una cadena de condiciones atómicas encerradas entre corchetes unidas con operadores lógicos (expresados mediante las etiquetas AND, OR, XOR), debiendo comenzar cada renglón con el operador lógico. Se deberán utilizar también los corchetes para romper la

precedencia de cálculo normal de estos operadores; por ejemplo: [[cond1] OR [cond2]] AND [cond3]. En términos generales, se podrá plantear una condición tan compleja como sea necesario, en la medida en que se respete la sintaxis de una expresión lógica.

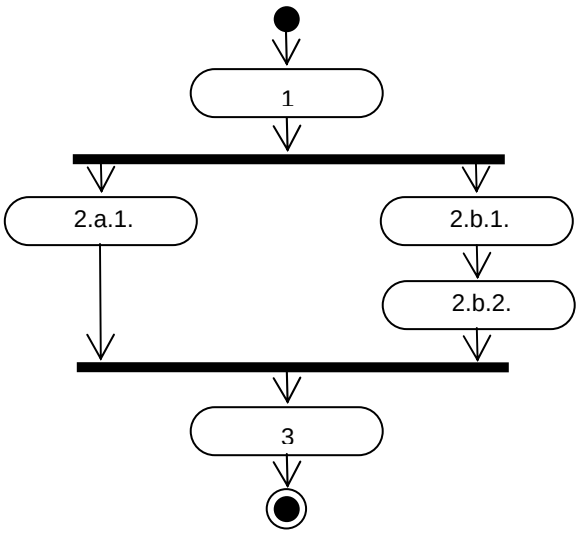
Pasos

Cada paso se debe redactar en forma una oración estructurada constituida por varias partes que, en conjunto, conforman una actividad. Cada paso debe tener asociado un número positivo que se incrementa de uno en uno a partir de la unidad.

Pasos Concurrentes

Cuando dos o más pasos se pueden realizar en un orden indistinto o en paralelo, estamos en presencia de "pasos concurrentes". Cada una de las ramas concurrentes se deberá distinguir mediante el agregado de una letra en letra minúscula, comenzando desde la letra “a” hasta la “z”. De esta manera, suponiendo que en un caso de uso los pasos número 3 y 4 pudieran ser intercambiados sin afectar su ejecución lógica, y además se considere útil o necesario dejar constancia expresa de esta situación, por el hecho de ser concurrentes ambos pasos tendrán la misma numeración, pero distinguiendo cada rama por medio de la letra recién mencionada. Así, los pasos 3 y 4 pasarán a ser 3a y 3b respectivamente.

Desde el punto de vista semántico, esta notación está reflejando concurrencia entre esos dos pasos, pero ambos deben completarse satisfactoriamente antes de pasar al paso siguiente (el ahora paso 4 que antes correspondía al 5). Gráficamente esta situación se representa por medio de las barras de sincronización. Por otra parte, cada una de las ramas puede contener más de un paso, para lo que deberá agregarse, luego de la letra y separada por el correspondiente punto de la numeración decimal, un nuevo dígito que se incrementará de uno en uno a partir de la unidad. En definitiva, los números siempre indicarán secuencia y las letras indicarán ramas concurrentes. A continuación podemos ver un ejemplo gráfico de lo dicho hasta el momento:



Nota: Como puede observarse en el diagrama, en caso de no contar con más de un paso por rama, no es necesario numerar la secuencia (3a).

Paso condicional

El paso condicional es un caso particular del subflujo (ver "subflujo" en el apartado "Curso Alternativo") basado en la premisa de lograr un curso normal muy claro y autocontenido (con la menor cantidad de referencias posibles). Un paso condicional es un subflujo que cuenta con un único paso. Para esto en el curso normal se debe indicar como sigue:

Nº de paso. SI [expresión lógica]
<paso>
SINO
<paso>

Ejemplo: SI [se encuentra habilitado] el Bibliotecario otorga el préstamo

Relación <<incluir>> (<<include>>)

Tal como establece el estándar UML, la asociación de dependencia estereotipada con <<incluir>> se emplea para evitar describir el mismo flujo de eventos repetidas veces, aportando un mecanismo de factorización que permite ubicar comportamiento común en un único caso de uso aparte que será compartido por todos aquellos casos de uso en los que originalmente se encontraba expresada la porción de pasos común. Semánticamente se debe interpretar que el caso de uso base "incluye" la funcionalidad expresada en el caso de uso común (por esa razón la flecha de la asociación de dependencia apunta al caso de uso común, puesto que el caso de uso base "depende" del aquél porque necesita su funcionalidad para poder completarse).

La relación de inclusión se expresa en un paso con el estereotipo <<incluir>> más el nombre del caso de uso común (o factorizado) entre paréntesis, en lugar del paso o de los pasos que fueron factorizados. Esta llamada genera el mismo efecto que insertar todo el caso de uso factorizado en el punto de inclusión indicado de esa manera.

Las asociaciones de extensión, que complementan al concepto de inclusión, se tratarán dentro del apartado "Curso Alternativo".

En algunas oportunidades, la complejidad inherente de la realidad que se estará tratando de modelar podría obligar al empleo de etiquetas que indican un cambio en la secuencia normal de ejecución del caso de uso. Estas etiquetas contendrán el número del paso del curso normal con el que se debe continuar. Este recurso debería ser utilizado exclusivamente en aquellas circunstancias en las que esta fuera la manera más clara de interpretar el desarrollo del caso de uso, pero sin dudas deberá ser utilizado en circunstancias excepcionales.

Las etiquetas, de acuerdo a lo establecido por el UML, son valores encerrados entre llaves. En nuestro caso, el valor será el prefijo “cn” (curso normal) seguido por el número de la secuencia del paso en la que debe continuar la ejecución del caso de uso; por ejemplo {cn1}.

Curso alternativo

La naturaleza propia de la mayoría de los procesos reales involucra condiciones que llevan a tomar diferentes cursos de acción, y expresar el desarrollo de un comportamiento incluyendo todas las alternativas posibles haría imposible la lectura del caso de uso. Dentro de estos pasos condicionales o cursos alternativos, encontramos dos tipos diferentes de pasos alternativos que, por su naturaleza, corresponden a dos situaciones diferentes que hemos decidido denominar:

- **Subflujo** (que se identificará con el prefijo "sf")
- **Excepción** (que se identificará con el prefijo "ex")

Una excepción se utiliza para describir funcionalidades indeseadas o excepcionales de un paso a causa de una condición no cumplida. Un subflujo se utiliza para expresar caminos alternativos de un paso en donde cada uno de los cursos de acción no es una excepción del otro, sino que todos tienen igualdad de importancia y son parte del camino normal.

De esta manera, las excepciones se deben documentar fuera del curso normal, mientras que los subflujos se deben documentar dentro del curso normal del caso de uso.

Subflujos

Cuando en un paso se debe tomar un curso de acción en función de una condición por la naturaleza propia del mismo problema o negocio, pero en donde las acciones no son excepcionales, estos cursos alternativos serán identificados como subflujos.

La condición de la que surgen los subfljuos se expresa como un paso en el curso normal, y los subflujos continúan como pasos sucesivos. Por ejemplo, en el caso de uso “Contratar Empleado” podría contratarse a una persona de otra empresa (curso normal ya que es el escenario mas frecuente), podría transferirse una persona de un departamento a otro (algo frecuente en ciertas compañías) o podría contratarse a un extranjero (conlleva sus reglas específicas). Estas dos últimas variantes o alternativas pueden expresarse como subflujos del curso normal.

El subflujo se representará con un código entre paréntesis en el curso normal anteponiendo el prefijo “sf” al número de paso en el que se encuentra la llamada, más un punto seguido por un número entero positivo que se incrementará de uno en uno comenzando con la unidad; por ejemplo (sf3.1). Debido a que los subflujos se ejecutan a partir de una condición en el curso normal, ésta se indicará como un paso condicional de la siguiente manera:

N° Paso SI [expresión lógica] {subflujo}
SINO {subflujo}

Ejemplo: 4 - SI [es extranjero] {sf3.1}
SINO {sf3.2}

La etiqueta SI y SINO deberán ir siempre en letra mayúscula. El empleo del SINO es opcional.

Relación <<extender>> (<<extend>>)

Cuando una condición en el curso normal de un caso de uso "A" implica la ejecución de un conjunto de pasos que se llevan a cabo en forma excepcional o que corresponden al tratamiento de un error, estos pasos excepcionales se deben agrupar y modelar como un nuevo caso de uso, caso de uso "B". De esta manera, el caso de uso base "A" será extendido por el caso de uso excepcional "B".

Entre los casos de uso "A" y "B" existirá una asociación de dependencia estereotipada como <<extender>>, lo que significa que la funcionalidad del caso de uso "B" "extiende" la funcionalidad del caso de uso "A" en tanto y en cuanto en el caso de uso "A" se cumpla la condición que implica la ejecución del caso de uso "B". Y la asociación de dependencia debe apuntar al camino base (contrariamente a lo que sucede en la asociación de inclusión) debido a que el caso de uso "B" depende de que se cumpla con una condición en "A" para poder llevarse a cabo.

Como se puede observar, la funcionalidad que estamos modelando con una asociación de extensión podría modelarse perfectamente también como un subflujo, por lo que se ha decidido establecer las siguientes reglas prácticas:

- un subflujo es, conceptualmente, diferente a una excepción, por lo que su tratamiento debería ser diferente: los subflujos son parte del camino base y las excepciones son tales, como su nombre lo indica.
- la funcionalidad de las excepciones debería tratarse como un caso de uso que extiende al caso de uso base, pero en algunas oportunidades esta funcionalidad excepcional es tan pequeña (en cuanto al número de pasos) que no sería conveniente, desde un punto de vista meramente práctico, destinar un caso de uso entero a esa pequeña porción de funcionalidad, describiéndose, entonces, en la sección destinada a los subfujos dentro del documento que contiene la descripción del caso de uso base.
- este criterio práctico será adoptado siempre que la excepción cuente con más de 10 pasos o que la excepción posea nuevas excepciones.

Es importante aclarar que las condiciones anteriores no representan una regla absoluta y automática ya que si el subflujo cumple con una o dos de las condiciones anteriores y el subflujo no puede considerarse, por el nivel de abstracción y criterio personal, un nuevo caso de uso extendido, puede dejarse como subflujo. En conclusión, estas reglas son para contar con un disparador que nos ponga en alerta de una posible relación de extensión.

En caso de requerir, bajo las condiciones anteriormente indicadas, generar una relación de extensión debe consignarse en el curso normal con la siguiente sintaxis:

Nº de paso. SI [expresión lógica] (código de caso de uso)
SINO (código de caso de uso)

Ejemplo: SI [el usuario está inhabilitado] (cu0039)

Asimismo en la sección destinada a los Subflujos, éstos deberán ordenarse de manera creciente por número de paso del curso normal y número de subflujo mediante el siguiente esquema:

Código de Subflujo	**Nombre del Subflujo**
Nº de paso	Paso

Un subflujo cuenta con la posibilidad de soportar excepciones indicándolas con el código utilizado para cualquier paso de un curso normal. Por ejemplo, {ex.sf.12.2.4} sería la cuarta excepción del segundo paso del subflujo llamado en el paso 12 del curso normal.

Excepciones

Una excepción es un curso alternativo que puede preverse pero no controlarse para ser evitado y que puede alterar el desarrollo del paso del curso normal. En general responde a situaciones no deseadas que alteran el curso normal.

Todos los pasos de un curso normal pueden realizarse con o sin éxito. En alguno de ellos es importante establecer las condiciones que determinan el éxito y en otros no. En aquellos pasos en donde es importante determinar la condición de éxito, cuando la ejecución del paso no fue exitosa debería indicarse la secuencia de pasos excepcionales que se deberían llevar a cabo como consecuencia del fracaso en la ejecución del paso del curso normal. En estos pasos los cursos alternativos que se generen se encontrarían originados por un desempeño “anormal” de una actividad o paso y no como un flujo alternativo propio de la naturaleza del problema.

Así, deberán enumerarse, separados por punto y coma, las diferentes excepciones que pueden presentarse en el desarrollo normal del paso, a continuación de él, entre llaves y en un renglón nuevo. Las excepciones se deben identificar mediante un código de excepción compuesto por el prefijo “ex” y seguido por el número del paso que produjo la excepción, más un punto seguido por un número entero positivo que se incrementará de uno en uno a partir de la unidad. Por ejemplo, “ex12.3” identifica a la tercera excepción que puede presentarse en el paso número 12 del curso normal. En la plantilla, el curso normal se vería de la siguiente forma:

12	<paso> {ex12.1;ex12.2;ex12.3}

Siguiendo con el ejemplo, para poder continuar con el paso siguiente (paso 13), deberán haberse evaluado cada una de las excepciones posibles. Las excepciones deberán ser ordenadas por número de paso y excepción, de menor a mayor y ser ubicadas dentro de la sección etiquetada como “Excepciones” de la plantilla:

Códigode Excepción	**SI [condición]**
Nº de paso	Paso

Cada excepción será documentada con el código, la condición de excepción (entre corchetes) y luego con idénticas características que un curso normal:

ex12.1	SI [el usuario no existe]
1 2	<paso> <paso>

4.3 Ejemplo A (Plantilla)

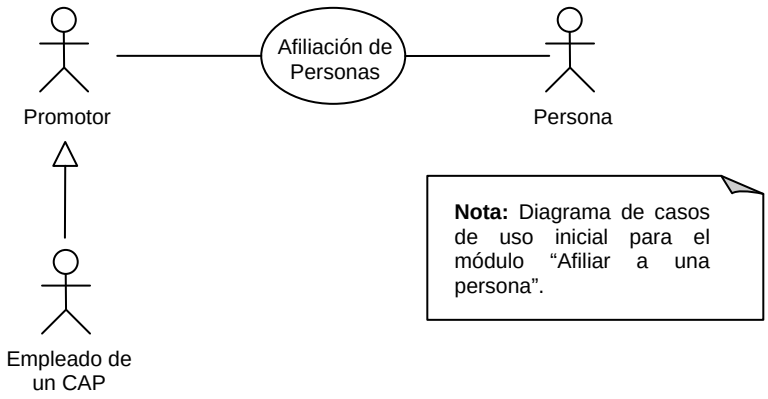
Código	**CU1**
Nombre	Hacer Reserva
Actores	Creador de Reserva, Sistema de Mensajería
Sinopsis	Este caso de uso comienza cuando el Creador de Reserva solicita crear una reserva. El sistema chequea la disponibilidad de una habitación en un hotel solicitado. Si hay disponibilidad el Sistema hace la reserva y le confirma la misma al cliente. Si no hay disponible una habitación, el sistema sugiere hoteles alternativos.
Precondiciones	[[cond1] OR [cond2]] AND [cond3].

Curso Normal de Eventos	
Nº	**Descripción**
1. 2. 3. 4. 5. 6	<<incluir>> (Identificar Cliente). Creador de Reserva indica hotel, tipo de habitación y duración de la estadía. Sistema confirma disponibilidad. Sistema registra la reserva. <<incluir>> (Confirmar Reserva). ……
Subflujos	
Nº	**Descripción**
sf6.1.	
1. 2.	
sf7.1.	
1. 2.	
Excepciones	
Nº	**Descripción**
ex12.1	
1. 2.	
ex15.1	
1. 2.	

4.4 Ejemplo B (Diagrama)

Estudiaremos un ejemplo concreto como para comprender mejor esta técnica. Tomaremos un caso simplificado, a fin de no perdernos en detalles, y lo suficientemente conocido como para que éste no sea un problema y nos permita comprender mejor la aplicación de la técnica de casos de uso. Consideraremos el proceso de afiliación de una persona a la organización.

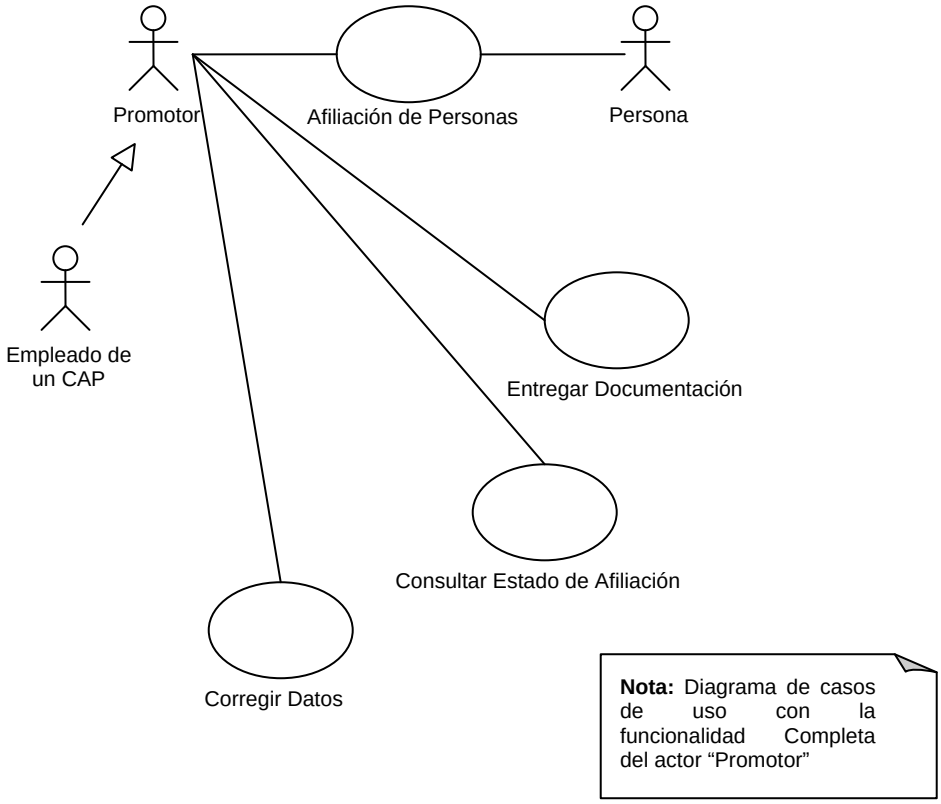
En este proceso seguramente encontraremos rápidamente como actores a la persona y al promotor. Podríamos pensar que si una persona es atendida en una oficina por acercarse a ella necesitaríamos otro actor diferente, pero lo que nos interesa aquí en primera instancia es el rol que cada actor cumple, y ya sea que se trate de un promotor o de un empleado de una oficina, el rol que cualquiera de ellos cumple al atender al afiliado es el de promotor. De esta manera, nuestro primer diagrama sería así:



El siguiente paso es elegir al actor más representativo, que en este caso podría ser "Promotor", y tratar de agotar toda la funcionalidad del sistema desde su punto de vista. En otras palabras, se le debe preguntar al promotor todo lo que él necesita del sistema, y su respuesta probablemente sería:

a) afiliar a una persona
b) entregar la documentación de afiliaciones para su procesamiento
c) consultar de vez en cuando el estado de la afiliación de sus clientes
d) visitar a sus clientes para corregir algún dato que pudiera haber quedado pendiente a fin de dar curso definitivo a la afiliación

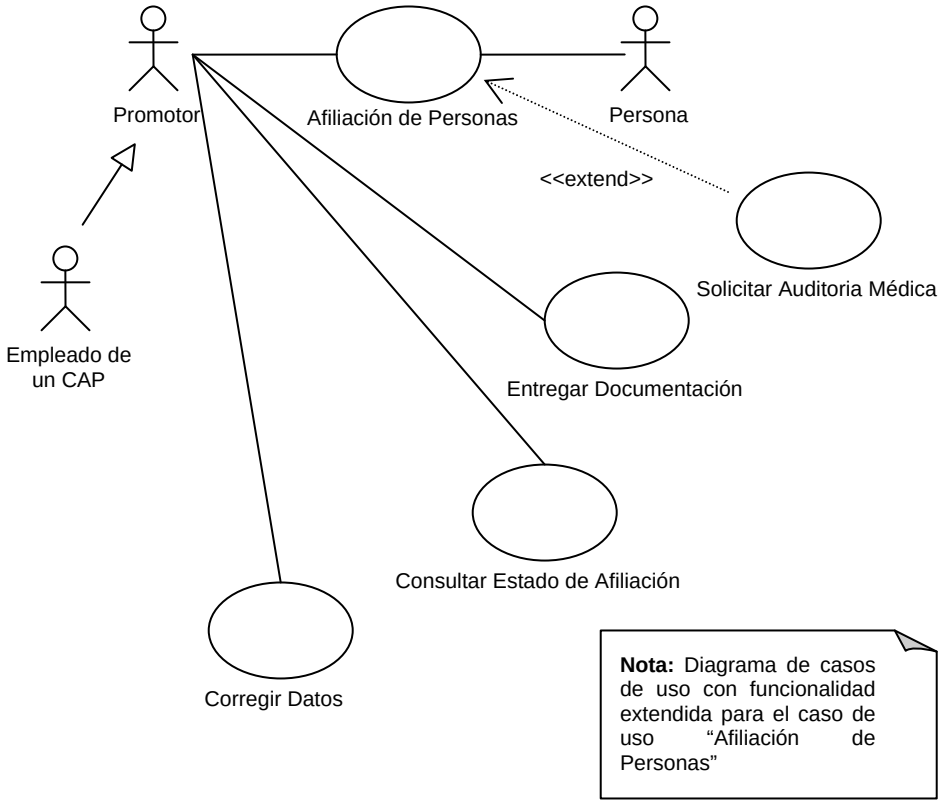
A fin de cuentas, el diagrama debería lucir de esta forma:



Cada uno de los casos de uso incorporados al diagrama debe poseer su documentación, es decir, la descripción detallada de qué pasos hay que dar para cumplir con el objetivo planteado por cada uno de ellos. Por ejemplo, el caso de uso "Afiliar a una persona" debería estar constituido por los siguientes pasos (Se destacaron en negrita los actores involucrados en el caso de uso, que coinciden con los establecidos en el diagrama):

1. **El Promotor** completa la solicitud de afiliación.
2. **El Promotor** hace firmar a **la Persona** las consideraciones generales.
3. **El Promotor** le entrega una copia a **la Persona**.
4. **La Persona** entrega la documentación de respaldo **al Promotor**.

Luego de describir todos los casos de uso, hay que tratar de detectar las excepciones o caminos alternativos que requieran una atención especial por el sistema, como así también la funcionalidad necesaria para el tratamiento de los errores. Por ejemplo, en el paso Nº 1, si el Promotor considera que es necesaria la participación de Auditoría Médica en la revisión de la solicitud, debe indicarlo en el formulario de afiliación. Por este motivo, el diagrama de casos de uso quedaría indicando que el caso de uso "Afiliar a una persona" debería extender su funcionalidad con la comprendida en el caso de uso "Solicitar participación de Auditoría Médica". Lo mismo debería realizar para cada paso de cada caso de uso, pero por razones de simplicidad presentamos sólo el ejemplo mencionado:



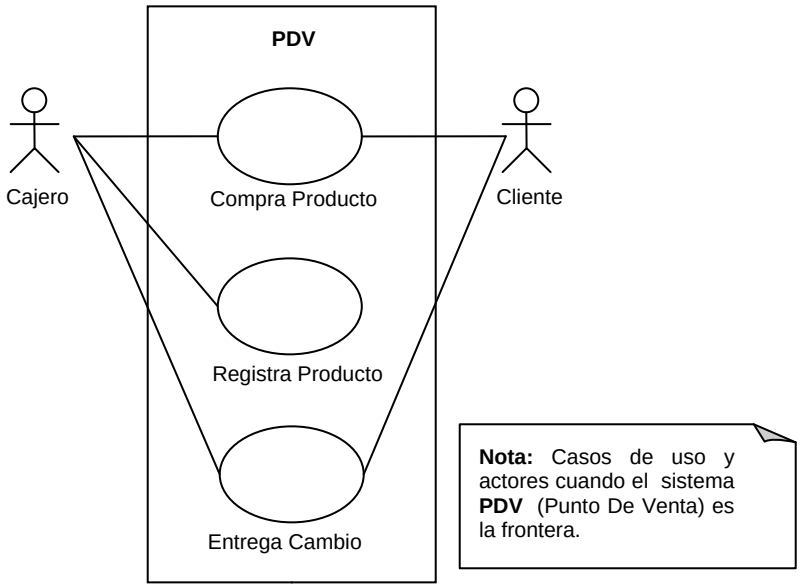
En definitiva, los casos de uso permiten organizar y sistematizar las tareas de relevamiento del analista de sistemas y potenciar radicalmente los resultados que puede obtener. Sin dudas que el uso de esta técnica impacta directamente en el modo en el que se deben realizar los relevamientos, como así también la forma en la que estos requerimientos contenidos en el modelo de casos de uso deben ser traducidos paulatinamente hasta conformar el código que interpretará una computadora. Los usuarios no deben preocuparse por los detalles técnicos de las herramientas, que deben quedar bajo la responsabilidad del analista, sino que, por el contrario, deben sentirse más tranquilos al saber que estará empleando una técnica que los considera como los principales protagonistas. Y los resultados reales que se obtienen con esta técnica son el argumento más contundente que se puede esgrimir a favor de su empleo.

4.5 Fronteras

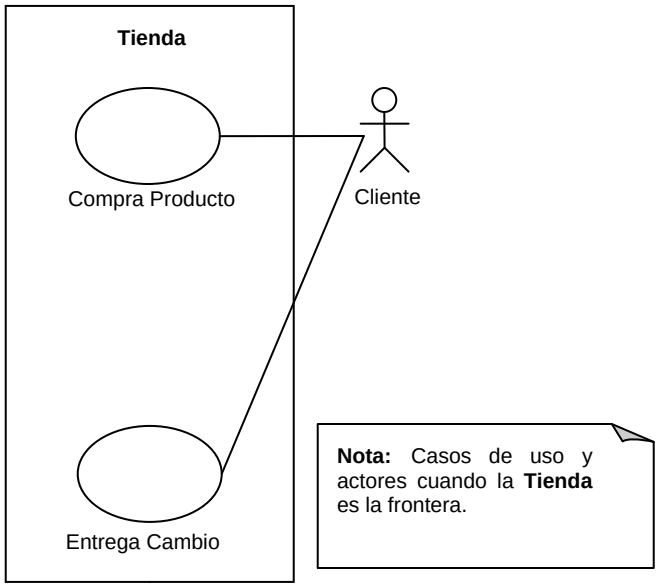
Un caso de uso define la interacción con un sistema. Las fronteras del sistema normalmente son: la frontera software/hardware de un dispositivo o sistema de cómputo, el departamento de una organización, la organización entera.

Las fronteras son importantes para definir lo que es interno y externo al sistema. El ambiente externo está representado exclusivamente por los actores.

4.5.1 Ejemplo A



4.5.2 Ejemplo B



4.6 Casos de Uso Esenciales y Reales

Casos de Uso Esenciales

- Casos expandidos que se expresan en forma teórica y con pocos detalles de implementación.
- Describe el proceso a partir de sus actividades y motivos esenciales.
- Conviene usarlos al comenzar a investigar los requisitos.
- Tienen una larga vida útil.

Casos de Uso Reales

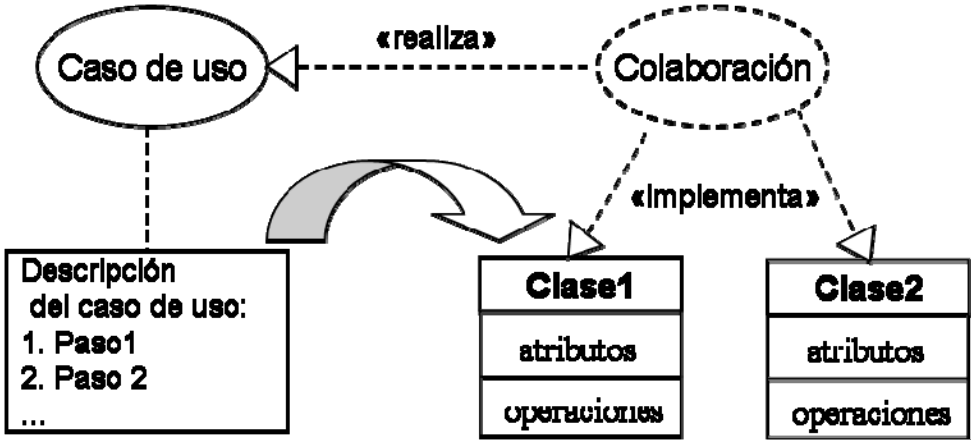
- Describen concretamente el proceso a partir de su diseño concreto actual, con sus tecnologías.
- Se usan principalmente durante el diseño.

4.7 Construcción de los Diagramas

- elaborar una lista de actores y definir sus roles.
- elegir el actor más representativo del sistema para comenzar el diagrama.
- agotar todas las necesidades funcionales del actor incorporando los casos de uso de la funcionalidad base.
- para cada caso de uso, buscar los actores que deban colaborar con él.
- repetir los dos pasos anteriores para cada actor.
- incorporar la funcionalidad necesaria para excepciones y errores.
- factorizar los casos de uso.
- obtener los actores abstractos mediante generalización.
- describir cada caso de uso a medida que se incluye en el modelo.
- validar y verificar el modelo junto con los usuarios.

4.8 Realización de los Casos de Uso (Use Case Realization)

Las acciones descriptas en un caso de uso deben ser ubicadas en objetos que colaboran entre sí para implementar la funcionalidad.



5 Modelo Conceptual (Domain Model)

Un modelo conceptual explica los conceptos más significativos en un dominio del problema, identificando los atributos y las asociaciones. Es la herramienta más importante del análisis orientado a objetos.

Un modelo conceptual representa cosas del mundo real, no componentes de software.

Un modelo conceptual es una descripción del dominio de un problema real, no es una descripción del diseño del software. Debido a esto, no es conveniente incluir elementos como ventanas o bases de datos.