Casos de Uso

Introducción

Los casos de uso son una técnica para analizar y especificar el comportamiento de un sistema o de una parte del mismo describiendo *qué* hace el sistema pero no *cómo* lo hace. Esta técnica fue introducida por Ivar Jacobson en 1992 y forma parte del estándar UML (Unified Model Language).
Si bien en un principio el modelado con casos de uso fue presentado para usarlo bajo la metodología de desarrollo orientada a objetos, en la actualidad, se utiliza sin importar la metodología elegida, tanto como parte del ciclo de vida de desarrollo del sistema como en el modelado ágil.
Los casos de uso capturan de manera natural y especifican los requisitos del sistema a desarrollar. Como sabemos, todos los sistemas ofrecen a aquellos que lo usan una serie de servicios o funcionalidades, en este contexto podemos decir entonces que un caso de uso es una forma de expresar como "algo o alguien" externo al sistema interactúa con el mismo usando dichas funcionalidades y obteniendo un resultado de interés para sí mismo. Cuando decimos "algo o alguien" estamos haciendo referencia a que los sistemas no solo son usados por personas, sino también por otros sistemas de hardware y/o software. En cualquier caso llamaremos "actor" a ese "algo o alguien" que interactúa con el sistema.

Veamos algunas definiciones….

Actores

¿Qué es un actor?

El actor es "algo o alguien" que usa el sistema, "algo" que usa el sistema puede ser otro sistema software o un sistema hardware, "alguien" que usa el sistema es representado por un rol particular que puede asumir una persona o un conjunto de personas al interactuar con el sistema que se está construyendo. Los actores *siempre* son externos al sistema, por lo tanto, al identificarlos se comienza a definir el límite y el alcance del mismo.

¿El actor, es el usuario?

Existe una diferencia entre el actor y el usuario, el actor representa un rol y el usuario es una persona o grupo de personas que cuando usa el sistema asume un rol, por lo tanto, puede ocurrir que una misma persona (usuario) en distintos momentos asuma roles diferentes y utilice el sistema de acuerdo al rol asumido. Por ejemplo, una persona que trabaja en un banco podría ser un *Ejecutivo de cuentas* y asumiría ese rol para interactuar con el sistema, pero a su vez, si esta persona tiene sus cuentas personales en ese banco estaría desempeñando también el rol de *Cliente*.

Tipos de actores

Como se ha mencionado, los actores no siempre representan personas, también puede representar sistemas software o sistemas hardware. Por ejemplo, si se está construyendo un *Sistema de ventas* que además de registrar las ventas genere los asientos contables para ser procesados por el *Sistema de contabilidad*, éste último sería un actor ya que es externo al sistema de ventas pero interactúa con él. Por otro lado, pensemos por ejemplo en un robot, si el sistema que se está construyendo contiene las rutinas necesarias para controlar sus movimientos, el *Robot* sería el actor que interactúa con el mismo. Otros ejemplos de este tipo de actores que representan sistemas hardware podrían ser un sensor, una impresora, un lector de código de barras, un postnet, etc.

¿Cómo se representan los actores?

Según el estándar UML 2.0, un actor se representa con un hombre de palo (stick man) con el nombre del rol que está cumpliendo debajo, o el nombre del sistema, si es que el actor está representando un sistema software o un sistema hardware.

Figura 1: Actor

Generalización de actores

¿Qué pasa si hay dos actores que hacen lo mismo? ¿Puede ocurrir?

Muchas veces puede ocurrir que un actor inicie todos los casos de uso que inicia otro actor y algunos más, por ejemplo, en un sistema de ventas, el *Supervisor de ventas* puede hacer todo lo que hace el *Empleado de ventas*, pero además puede autorizar las ventas, en este caso se puede decir que el supervisor de ventas *hereda* al empleado de ventas, o que el empleado de ventas *generaliza* al supervisor de ventas, de esta forma toda la funcionalidad que está habilitada para el empleado de ventas también lo está para el supervisor. Usando la generalización de actores se evita la redundancia y se simplifica la especificación y el diagrama de casos de uso al no tener que trazar tantas relaciones entre actores y casos de uso.

Figura 2: Generalización de actores

Ejemplo:

Como se puede ver en siguiente ejemplo, la herencia de actores se representa mediante una relación de generalización, y la relación entre los actores y los casos de uso mediante una asociación.

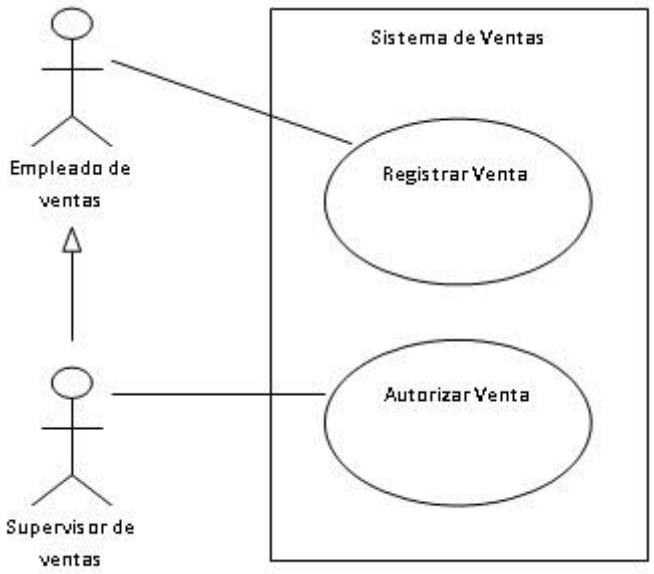

Figura 3: Ejemplo de generalización de actores

Casos de Uso

¿Qué es un Caso de Uso?

Un caso de uso es iniciado por un único actor a la vez y representa una funcionalidad completa del sistema que se describe mediante la secuencia de interacciones entre el sistema y el actor que inicia el caso de uso. La ejecución de un caso de uso debe dejar un resultado observable y de interés para el actor que lo inició.
Los casos de uso se utilizan para especificar los requisitos funcionales del sistema, en cada caso de uso, la interacción entre el actor y el sistema involucra todos los pasos necesarios para llevar a cabo una funcionalidad dada describiendo *qué* hace el sistema pero sin especificar *como* lo hace. Solo se pueden ver las entradas y las salidas del sistema, pero nunca los procesos. Cuando se describe un caso de uso se debe especificar el escenario principal, es decir, la secuencia de interacciones que sigue el curso normal de los acontecimientos, y las alternativas. Las alternativas son errores o eventos excepcionales que conforman otro escenario dentro de la descripción del caso de uso.

Escenarios

Un escenario es una secuencia específica de acciones que describe un comportamiento. La descripción de un caso de uso define uno o más escenarios, el escenario principal, que describe el curso normal de los acontecimientos y uno o más escenarios secundarios donde se describen las alternativas. Rumbaugh, Booch y Jacobson en su libro UML 2.0 (Rumbaugh, Jacobson, & Booch, El Lenguaje Unificado de Modelado UML 2.0, 2da ed. 2006) plantean que los escenarios son a los casos de uso lo que las instancias a las clases, es decir, un escenario es básicamente una instancia de un caso de uso.

¿Como se representan los casos de uso?

Según el estándar UML 2.0, un caso de uso se representa con un ovalo que lleva el nombre del caso de uso en su interior[1]. El nombre de un caso de uso se expresa desde el punto de vista del actor que lo inicia y está compuesto por un verbo en infinitivo seguido del objeto o entidad del sistema que es afectada por el mismo, por ejemplo, *Registrar venta*, *Anular pedido*, *Autorizar venta*, etc.

Figura 4: Caso de uso

Características de los casos de uso

Los casos de uso tienen las siguientes características:

- Están expresados desde el punto de vista del actor y no del sistema, el punto de vista del actor representa una vista externa del sistema, por ejemplo, el caso de uso *Ingresar pedido* iniciado por el actor *Empleado de ventas*, permite que el empleado de ventas ingrese un pedido en el sistema de ventas (visto desde su punto de vista), pero si se ve desde el punto de vista del sistema, el mismo caso de uso podría llamarse *Recibir información de pedido*.

- Se documentan con texto informal, en primera instancia, el documento que detalla la especificación de los casos de uso va a estar dirigido al cliente, por lo tanto, se debe documentar utilizando un lenguaje natural para que el cliente lo entienda.

- Describen tanto lo que hace el sistema como lo que hace el actor cuando interactúa con él.

- Son iniciados por un único actor a la vez, un mismo caso de uso puede ser iniciado por más de un actor en momentos diferentes, pero no puede haber casos de uso que no sean iniciados por ningún actor.

- Están acotados a una sola funcionalidad completa del sistema, una funcionalidad completa es un caso de uso si se debe indicar explícitamente al sistema que se quiere acceder a dicha funcionalidad. Por ejemplo, si se quiere registrar una venta se debe acceder a la funcionalidad *Registrar Venta*, en cambio, si solo se quieren ingresar los artículos vendidos para una venta en

[1] Si bien están basadas en el estándar UML, algunas herramientas de modelado colocan el nombre del caso de uso debajo del ovalo en lugar de en su interior, por lo tanto, las variaciones en la notación dependerán de la herramienta utilizada a la hora de modelar.

particular no debemos indicárselo explícitamente al sistema, simplemente esa es una funcionalidad que forma parte de una funcionalidad mayor que es *Registrar Venta*, por lo tanto, no sería un caso de uso.

- Los casos de uso deben dejarle un beneficio o resultado de interés al actor que lo inició, por ejemplo, el beneficio que obtiene el *Empleado de ventas* al ejecutar el caso de uso *Registrar venta*, es que la venta queda registrada.

Modelo de Casos de Uso

El modelo de casos de uso (MCU) presenta al sistema desde la perspectiva del actor y facilita la comunicación entre el cliente y el equipo de desarrollo para llegar a un acuerdo sobre los requisitos, condiciones y posibilidades que debe cumplir el sistema. Luego de la primera revisión del cliente, este modelo se irá refinando hasta llegar a su versión final que servirá como acuerdo o contrato donde se especifican todas las funcionalidades que el sistema debe cumplir y el equipo de desarrollo se compromete a desarrollar.

El MCU está compuesto por:

- El diagrama de casos de uso.

- La descripción de los casos de uso.

Diagrama de Casos de Uso

El diagrama de casos de uso está compuesto por un rectángulo que representa y delimita del sistema dentro del cual debe estar el nombre del mismo, los casos de uso, relaciones entre actores y casos de uso y relaciones entre casos de uso. Los actores son externos al sistema, por lo tanto, siempre se dibujan fuera del rectángulo y se representan con el gráfico correspondiente según sea su rol y el nombre que debe indicar qué rol está cumpliendo ese actor. Dentro del rectángulo estará lo que hace el sistema, todas sus funcionalidades representadas por casos de uso, cada caso de uso se va a representar con un óvalo que lleva su nombre en el interior, recordando que el nombre debe comenzar con un verbo en infinitivo. En el diagrama se puede ver claramente cuales son los casos de uso iniciados por cada actor mediante la asociación que los une. Si existieran, también se deben graficar las relaciones de generalización entre actores y las relaciones de inclusión y extensión entre casos de uso, estas últimas al estar relacionando funcionalidades, se dibujan siempre dentro del rectángulo que representa al sistema. Las relaciones entre casos de uso se explicarán más adelante dentro de este capitulo.

Ejemplo:

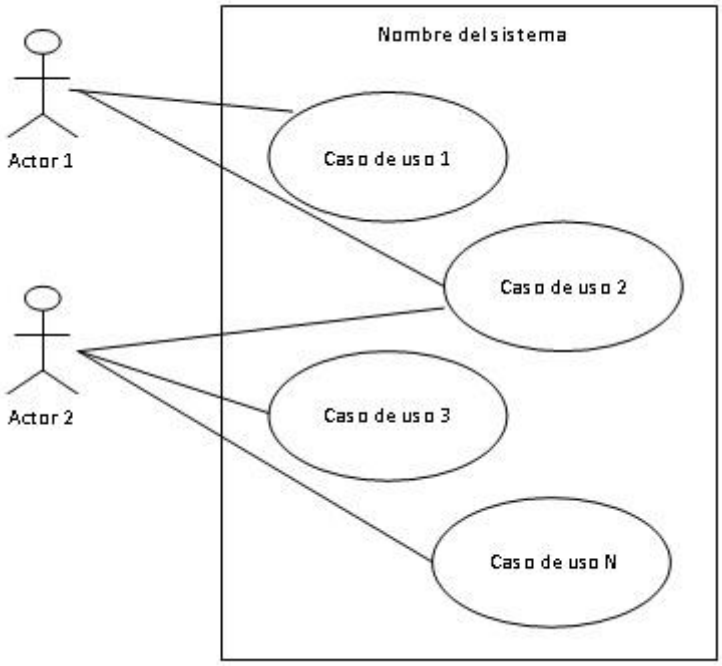

Figura 5: Diagrama de casos de uso

Descripción de los Casos de Uso

Luego de realizar el diagrama se debe hacer una descripción completa de cada uno de los casos de uso incluidos en el mismo. Para describir los casos de uso se utiliza una plantilla en la cual hay que indicar:

- **Nombre del caso de uso**: comienza con un verbo en infinitivo y está expresado desde el punto de vista del actor.
- **Actor principal**: es el que inicia el caso de uso y el que interactúa con el sistema.
- **Actor secundario**: participa una vez iniciado el caso de uso, los actores secundarios pueden ser varios o ninguno y no van a estar en el diagrama.
- **Descripción general o trazo grueso**: es una breve descripción, a grandes rasgos, de lo que hace el caso de uso, donde se debe especificar quién es el actor involucrado, la funcionalidad que se va a llevar a cabo y cual es el beneficio que deja el caso de uso para el actor.
- **Precondición**: como precondición se debe indicar el nombre del/los casos de uso que tienen que haberse ejecutado antes del caso de uso que se está describiendo. Por ejemplo, si existe un caso de uso *Registrar Venta* de productos, la precondición debe ser *Registrar Productos*.
- **Postcondición**: como postcondición se indica cual es el resultado de interés o beneficio que deja el caso de uso para al actor.

- **Descripción detallada o Trazo fino**: es la descripción de la interacción entre el actor y el sistema, se debe indicar paso a paso qué es lo que hace el actor y qué responde el sistema, sólo se indica la respuesta del sistema (lo que muestra), no lo que hace, por ejemplo, no se incluye en esta descripción si el sistema hace cálculos, realiza búsquedas, etc. También es común, y resulta muy intuitivo, acompañar esta descripción detallada con un prototipo de interfaz tentativa donde se puedan seguir los pasos de la interacción descripta.

  La descripción detallada de la interacción se divide en la descripción del *curso o flujo normal* y la descripción del *curso o flujo alternativo*:

  - **Flujo normal**: el flujo normal se describe en base a la interfaz y sigue el curso normal de los acontecimientos, es el caso en el que "todo sale bien". Dentro del flujo normal puede ocurrir que el actor tenga que realizar varias interacciones generando una serie de pasos que se repiten, por ejemplo, seleccionar varios productos para una venta. Estos casos se suelen expresar con frases como "para cada *x* seleccionado se repiten los pasos *y* a *z*" o "se repite el paso *x* hasta que ocurra *y*", etc.

  - **Flujo alternativo**: en el flujo alternativo se describen las alternativas al flujo normal. Dentro de la interacción puede haber alternativas a las acciones del actor y/o a las respuestas del sistema, pero cabe destacar que, en general, las alternativas son a las respuestas del sistema, es decir, ante determinada acción del actor se espera que el sistema responda de una manera y lo hace de otra, en estos casos además de describir la alternativa hay que indicar a que punto del flujo normal se vuelve luego o si se finaliza el caso de uso. Por ejemplo, sabemos que durante la ejecución de un caso de uso pueden aparecer errores o excepciones que se tratarán como alternativas, si se está ejecutando el caso de uso *Registrar Venta* y el actor en forma errónea ingresa un código de producto inexistente, en este caso el sistema deberá informar de dicha situación y regresar al punto del flujo normal donde se permita al actor ingresar nuevamente el código de producto.

    Las alternativas son entonces desviaciones del flujo normal y tienen las siguientes características:

    - Representan un error o una excepción.
    - No tienen sentido en sí mismas fuera del contexto del caso de uso en el que ocurren.

  Si bien es muy común ver que las alternativas se documentan al final del flujo normal, también resulta muy útil documentar los casos de uso en tablas, mostrando el flujo normal en la primera columna y el flujo alternativo en la segunda, de esta manera es mucho mas simple ver en qué paso del caso de uso puede ocurrir una excepción, teniendo la ventaja de poder leer de corrido el flujo normal.

Relaciones entre Casos de Uso

Existen dos tipos de relaciones entre casos de uso:

- Relaciones de extensión (extend)
- Relaciones de inclusión (include)

Estas relaciones se podrán detectar una vez hecha la descripción detallada de todos los casos de uso.

Relaciones de extensión: en una relación de extensión, un caso de uso que extiende a otro le agrega funcionalidad al caso de uso extendido y se ejecutará de manera *opcional* cada vez que se ejecute éste. La funcionalidad descripta en el caso de uso *"que extiende"* representa una serie de pasos opcionales o una excepción dentro de la funcionalidad descripta en el caso de uso *"extendido"* que se ejecutará sólo en algunas oportunidades. Por ejemplo, supongamos que estamos describiendo el caso de uso *Registrar venta*, puede ocurrir que tengamos que vender a un cliente nuevo que no esté registrado en el sistema, en este caso aparece una excepción, la excepción consiste en interrumpir la ejecución de este caso de uso y pasar a la ejecución del caso de uso *Registrar cliente*, de esta manera podemos decir que el caso de uso *Registrar cliente* que se ejecuta de manera opcional ***extiende*** o agrega funcionalidad al caso de uso *Registrar venta*.

Las relaciones de extensión entre casos de uso se representan en el diagrama con una relación de dependencia desde el caso de uso "*que extiende a"* hacia el caso de uso *"extendido"* acompañada por el estereotipo UML <<extends>>.

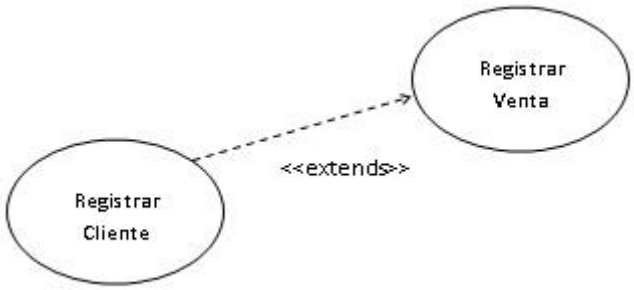

Figura 6: Relación de extensión

Además, las relaciones de extensión tienen las siguientes características:

- Representan una parte de la funcionalidad de un caso de uso que *no siempre* ocurre.

- La funcionalidad *"que extiende"* al caso de uso también es un caso de uso en sí mismo.

- Si bien es lo más común, no necesariamente provienen de un error o una excepción.

Diferencias entre una alternativa y una relación de extensión:

- En una relación de extensión, la funcionalidad que extiende a un caso de uso es un caso de uso en sí mismo mientras que una alternativa no lo es.

- Una alternativa es un error o excepción, mientras que una extensión puede no serlo.

Relaciones de inclusión: luego de realizar la descripción detallada de los casos de uso, puede ocurrir que se detecte alguna serie de pasos que se repite en dos o más casos de uso. Si esta serie de pasos representa una funcionalidad completa, a partir de ella se crea un nuevo caso de uso que estará *"incluido"* en cada uno de los casos de uso que utilicen dicha funcionalidad y que se ejecutará *siempre* que éstos se ejecuten. Por ejemplo, la funcionalidad *Buscar producto* puede ser accedida por los casos de uso *Registrar pedido*, *Registrar venta*, etc., se puede entonces, crear el caso de uso *Buscar producto* y relacionarlo a través de una relación de inclusión con los casos de uso que lo utilizan. De esta manera, *Buscar producto*, al ser el caso de uso ***incluido*** se ejecutará siempre que se ejecuten los casos de uso que lo incluyen.

Las relaciones de inclusión se representan en el diagrama con una relación de dependencia desde el caso de uso *"que incluye"* hacia el caso de uso *"incluido"* acompañada por el estereotipo UML <<include>>.

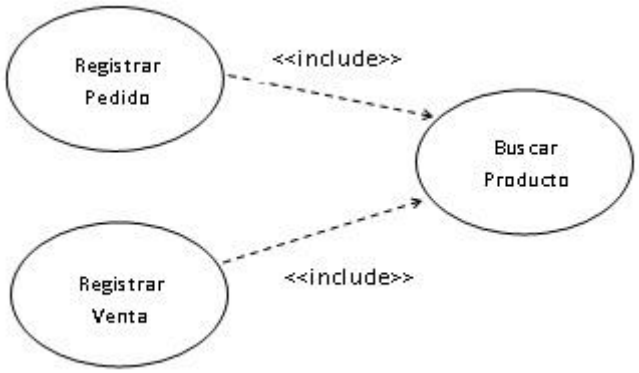

Figura 7: Relación de inclusión

Las características de las relaciones de inclusión son:

- Aparecen como una funcionalidad común luego de haber especificado varios casos de uso.

- Las funcionalidades incluidas son casos de uso en sí mismos.

- El caso de uso incluido se ejecuta *siempre* que el caso de uso que lo incluye se ejecute. Esta es la principal diferencia con las relaciones de extensión donde un caso de uso que extiende a otro se puede ejecutar, o no, de manera opcional.

Para que exista una relación de inclusión tiene que haber por lo menos *dos casos de uso* que incluyan a un tercero (recordemos que los casos de uso que son incluidos surgen a partir de los pasos comunes que se repiten en la descripción de varios casos de uso), o, que el caso de uso incluido sea accedido por un caso de uso y por el actor directamente.

Ejemplos

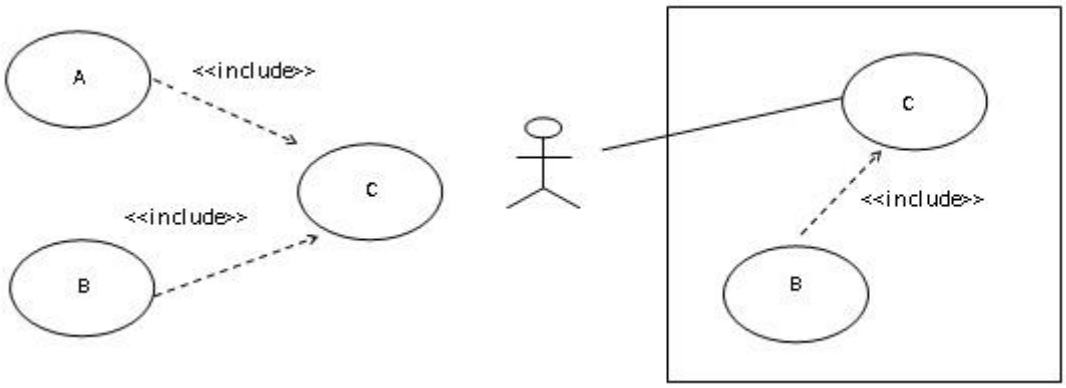

Figura 8: Ejemplos de include

A continuación, se muestra un caso donde no hay relación de inclusión ya que el caso de uso *C* no es accedido por un segundo caso de uso ni tampoco directamente por el actor, por lo tanto *C* simplemente es una parte de la funcionalidad de *B*.

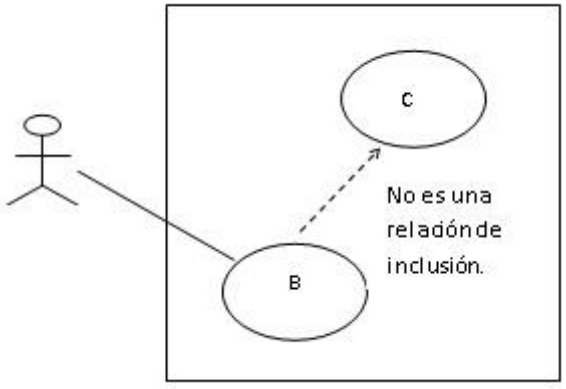

Figura 9: Ejemplo donde no existe relación de inclusión

Hasta que no se complete la descripción detallada de los casos de uso no se pueden identificar estas relaciones ya que no se habrán encontrado funcionalidades comunes u opcionales. Luego, las relaciones de extensión aparecen en el flujo alternativo ya que llaman a la ejecución de un caso de uso de manera alternativa u *opcional*. En cambio, las relaciones de inclusión aparecen dentro del flujo normal ya que *siempre* llaman a la ejecución de un caso de uso de manera obligatoria como parte del curso normal de los acontecimientos.

Metodología de trabajo

Rumbaugh, Booch y Jacobson en su libro "*El proceso unificado de desarrollo de software*" (Rumbaugh, Jacobson, & Booch, El Proceso Unificado de Desarrollo de Software, 1ra ed. 2000) plantean una metodología de trabajo para realizar en forma correcta el MCU que consiste básicamente en encontrar los actores que van a interactuar con el sistema, encontrar los casos de uso para cada actor y describirlos primero brevemente y luego en forma detallada para completar el modelo. Identificar actores y casos de uso es una actividad muy importante ya que ayuda a delimitar el sistema y a definir su alcance. Si bien esta actividad es responsabilidad del analista de sistemas, requiere cierta participación del cliente ya que este último es quien proporciona los requerimientos y la información a cerca de los posibles usuarios del sistema.
El desarrollo del MCU es iterativo, el analista de sistemas, junto a su equipo, pueden crear un modelo preliminar que luego se validará con el cliente quien podría sugerir modificaciones o agregados de funcionalidades nuevas, de esta manera se va refinando el modelo hasta llegar a la versión definitiva que será aprobada por el cliente y sirve como contrato donde se especifican todas las funcionalidades que el sistema debe cumplir.

Aplicar esta metodología consiste en:

1) **Identificar actores**: antes de avanzar con los casos de uso se debe tratar de identificar a todos los actores que van a interactuar con el sistema. El analista de sistemas junto con el cliente tiene la tarea de identificar los usuarios y organizarlos según el rol que van a representar al interactuar con el sistema, esta tarea permite definir una lista de actores candidatos. En un primer momento es común encontrar varios actores que cumplan con el mismo rol, en estos casos habrá que revisar la lista de actores candidatos para finalmente encontrar el conjunto de actores adecuados y eventualmente definir generalizaciones en los casos necesarios. El paso siguiente será asignarle a cada actor un nombre que defina claramente el rol que esta representando y describir brevemente para que utilizará el sistema, indicando necesidades y responsabilidades.

   Como resultado de este paso se obtiene un conjunto de actores actualizado, cada uno con su descripción, que pueden utilizarse como punto de partida para encontrar los casos de uso.

2) **Identificar los casos de uso para cada actor**: el analista de sistemas irá revisando los actores y proponiendo los casos de uso para cada uno, estos casos de uso propuestos en primera instancia son casos de uso candidatos y no todos llegarán a ser casos de uso por sí mismos. Recordemos que un caso de uso debe representar una funcionalidad completa del sistema y dejar un resultado de valor para el actor, revisar minuciosamente los casos de uso candidatos es una tarea importante ya que este análisis permite identificar las funcionalidades completas que representan casos de uso en si mismos y las que

son sólo parte de otro caso de uso. Una vez encontrados los casos de uso para cada actor se le debe asignar un nombre a cada uno, el nombre de un caso de uso siempre comienza con un verbo en infinitivo y debe reflejar cual es el objetivo de la interacción entre el actor y el sistema visto desde el punto de vista del actor, por ejemplo, un caso de uso llamado *Registrar venta* refleja claramente que el objetivo del *Empleado de ventas* (actor) al interactuar con el sistema es que la venta quede registrada en el mismo.

Como resultado de este paso se obtiene un primer diagrama de casos de uso sujeto a evaluaciones y restructuraciones.

3) **Identificar nuevos casos de uso a partir de los existentes**: en este paso se debe verificar si el sistema está completo o completarlo buscando otras funcionalidades a partir de:

    a. *Variaciones significativas de los casos de uso existentes*: se pueden encontrar nuevos casos de uso a partir de las variantes de un caso de uso existente. Por ejemplo, si existieran muchas diferencias entre registrar la venta a un cliente minorista y registrar la venta a un cliente mayorista, entonces, del caso de uso *Registrar Venta* se podrían obtener los casos de uso *Registrar venta a cliente minorista* y *Registrar venta a cliente mayorista*, siempre y cuando las variaciones del caso de uso lo justifiquen.

    b. *Casos de uso opuestos*: se podrían encontrar nuevas funcionalidades para completar el sistema buscando casos de uso opuestos a los existentes, por ejemplo, si existe el caso de uso *Registrar venta*, debería existir también el caso de uso *Anular venta*.

    c. *Casos de uso que preceden a casos de uso existentes*: siempre hay que preguntarse que sucede antes del caso de uso que se está analizando. Por ejemplo, si se está registrando una venta a un cliente, el cliente ya tendría que estar registrado como tal en el sistema, por lo tanto, debería existir otro caso de uso llamado *Registrar Cliente*.

    d. *Casos de uso que suceden a los casos de uso existentes*: en este caso hay que preguntarse que ocurre después del caso de uso que se está analizando. Por ejemplo, ¿qué ocurre después de *Registrar Venta*?, ésta es una buena forma de asegurarse que no se están dejando requerimientos sin identificar y permite descubrir partes de alguna nueva funcionalidad y nuevos casos de uso.

Luego de analizar todos estos casos, se obtiene como resultado un segundo diagrama de casos de uso más completo donde se agregan los nuevos casos de uso encontrados.

4) *Describir los casos de uso con trazo grueso*: una vez identificados todos los casos de uso hay que empezar a documentarlos haciendo la descripción con trazo grueso. Recordemos que ésta debe ser una breve descripción, realizada desde la perspectiva del actor, que especifique lo que hace el caso de uso indicando quien es el actor involucrado, la funcionalidad que se va a llevar a cabo y cual es el resultado de valor que deja ese caso de uso para el actor.

   Como resultado de este paso se obtiene el MCU compuesto por el diagrama de casos de uso completo y la descripción con trazo grueso de cada caso de uso.

   Este MCU representa los requerimientos del cliente "convertidos" en casos de uso que especifican las funcionalidades que el sistema debe cumplir junto a sus descripciones y a los actores que van a interactuar con las mismas.
   Llegado este punto, el cliente deberá ver el MCU para determinar si los casos de uso capturan todos sus requerimientos, si la descripción de los mismos es correcta, completa y comprensible y si identifica algún caso de uso que no proporcione ningún resultado de valor, en cuyo caso habría que reconsiderar la inclusión del mismo en el modelo.
   Es muy importante que el analista de sistemas, junto con su equipo, revisen bien éste modelo antes de presentárselo al cliente ya que el mismo servirá como contrato donde se especifica todo lo que el sistema "garantiza que cumplirá" si el cliente decide aceptar y se sigue adelante con el proyecto.

5) *Priorizar los casos de uso*: en este paso se deben definir prioridades y seleccionar los casos de uso de la primera iteración. Una vez documentados los casos de uso con el trazo grueso es conveniente definir las prioridades de los mismos, para determinar cuáles serán tenidos en cuenta en las primeras iteraciones del proceso de desarrollo y cuáles se pueden dejar para más adelante. Sabiendo que los casos de uso especifican los requisitos funcionales del sistema basados en los requerimientos del cliente, para priorizarlos se debe tener en cuenta la siguiente categorización:

   a. Requerimientos *imprescindibles o de prioridad alta*, son aquellos requerimientos que si no se implementan el sistema no funcionaría y no tendría sentido.

   b. Requerimientos *importantes o de prioridad media*, son aquellos requerimientos muy deseables, pero no imprescindibles y se pueden negociar con el cliente.

   c. Los requisitos *deseables o de prioridad baja*, son aquellos requerimientos posibles de implementar pero que podrían eliminarse. Solo se implementan si hay tiempo disponible.

   Una vez categorizados los casos de uso, se seleccionan los que se van a desarrollar en la primera iteración, en general, se incluyen los de prioridad alta,

se discuten los de prioridad media y se descartan los de prioridad baja cuyo costo sea alto.

Como resultado de este paso se obtienen los casos de uso priorizados para comenzar a planificar cuáles se van a desarrollar en cada iteración del proceso de desarrollo.

6) *Describir los casos de uso con trazo fino*: una vez seleccionados los casos de uso que se van a implementar en la primera iteración, se realiza la descripción detallada de cada caso de uso escribiendo el trazo fino, recordemos que la descripción detallada es la descripción paso a paso de la interacción entre el actor y el sistema, siempre realizada desde el punto de vista del actor, y se divide en el flujo normal, que muestra el curso normal de los acontecimientos, el caso en "que todo sale bien" y el flujo alternativo que muestra las desviaciones del curso normal por posibles errores o excepciones. Es muy importante describir todas las alternativas o de lo contrario no se estaría especificando por completo el caso de uso, estas desviaciones del curso normal, errores o excepciones pueden ocurrir por varias razones, por ejemplo, el actor puede elegir entre diferentes caminos en el caso de uso, sólo uno de esos caminos (el que habitualmente se va a seguir) se debe describir en el curso normal y los demás se describen como alternativas. Otro ejemplo de errores o excepciones se puede dar cuando el sistema detecta una entrada errónea por parte del actor, lo cual también se debe contemplar como una alternativa al curso normal.
   En este paso se crea un prototipo de interfaz del usuario para cada caso de uso a describir que ilustre cómo el actor puede ejecutarlo y brinde una idea de como se verá el sistema. Teniendo una interfaz tentativa donde se visualicen los datos involucrados en la ejecución del caso de uso resulta más sencillo escribir la descripción detallada de la interacción indicando las acciones del actor y las respuestas del sistema.

   Como resultado de este paso se obtiene la descripción detallada de los casos de uso junto a sus prototipos de interfaz.

7) *Identificar relaciones entre casos de uso*: luego de describir los casos de uso con trazo fino, revisando estas descripciones, se pueden detectar secuencias de pasos que se repiten *siempre* en distintos casos de uso. Para evitar este tipo de redundancia, si no existiera, se debe crear un nuevo caso de uso que describa esta secuencia de pasos repetitiva y relacionarlo con los casos de uso que incluyen dicha secuencia a través de una relación *include*, de esta manera cada vez que se ejecute un caso de uso A que incluya al caso de uso B, también se ejecutará B. Ahora bien, si se encuentra una secuencia de pasos que se repiten en distintos casos de uso, pero solo en algunas oportunidades, dicha secuencia sería *opcional* o alternativa dentro de la ejecución de un caso de uso dado, y en caso de ejecutarse, estaría extendiendo o añadiendo funcionalidad al caso de uso en cuestión. En estos casos, si no existiera, se crea un nuevo caso de uso que describa la secuencia opcional y se lo relaciona con los casos de uso

extendidos a través de una relación *extend*. De esta manera un caso de uso A que extiende a un caso de uso B se puede ejecutar o no cada vez que se ejecute B, la llamada a la ejecución del caso de uso A se hará en algún punto del flujo alternativo dentro de la descripción detallada del caso de uso B, una vez finalizado el caso de uso A se continua con la secuencia descripta dentro del trazo fino del caso de uso B.
Así se puede ver claramente la diferencia entre las relaciones extend e include, a diferencia del caso anterior, si un caso de uso A incluye al caso de uso B, la llamada a la ejecución del caso de uso B se hará en algún punto dentro del flujo normal de la descripción detallada del caso de uso A ya que B se ejecutará siempre que se ejecute A.

Como resultado de este paso se obtiene el diagrama de casos de uso completo incluyendo las relaciones entre casos de uso.

En resumen, al aplicar esta metodología de trabajo se obtiene el MCU que captura los requisitos del sistema y que está compuesto por el diagrama de casos de uso, la descripción general y la descripción detallada de cada caso de uso. Utilizando el proceso unificado de desarrollo de software como marco de trabajo, el MCU obtenido será el punto de partida para completar las etapas de análisis, diseño, implementación y prueba ya que los casos de uso van a dirigir el trabajo a lo largo de estas etapas iteración por iteración.

Casos de estudio

A continuación, se plantean dos casos de estudio de los cuales se desarrollará el modelo de casos de uso:

1. Sistema de alquiler de películas para un video club
2. Sistema de reservas de habitaciones para una cadena hotelera

Sistema de alquiler de películas para un video club

En un videoclub cuando una persona quiere alquilar una película se tiene que hacer primero socio (dando sus datos personales: DNI, apellido y nombre, domicilio, teléfono, celular, mail) luego le indicara al empleado de alquileres la película que desea. Se debe tener en cuenta si hay ejemplares disponibles.
Nota: se alquila *una sola película*.

Desarrollo del modelo de casos de uso

Actores: Empleado del video club, Encargado del video club

Casos de uso para cada actor:

Empleado del video club: Registrar socio, Registrar alquiler.

Encargado del video club: Actualizar datos socio, Eliminar socio, Generar listado de alquileres vencidos.

Diagrama de casos de uso[1]:

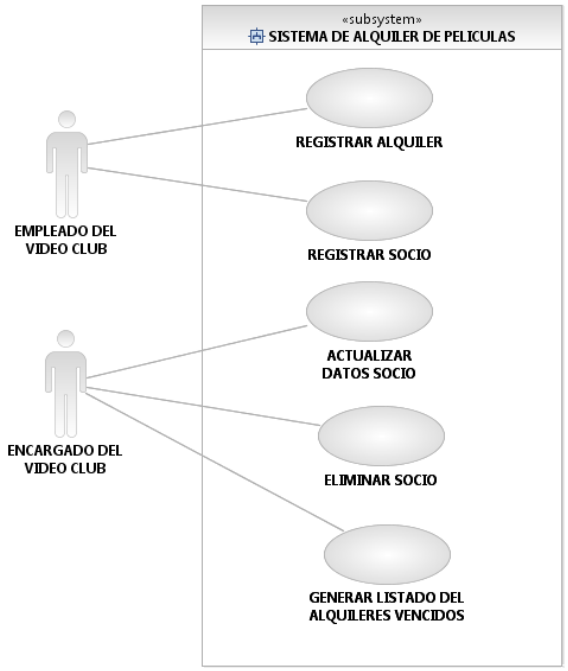

Figura 10: Diagrama de casos de uso del sistema de alquiler de películas

[1] Realizado con la herramienta IBM ® Rational® Software Architect™

Descripción de los casos de uso:

Registrar socio: el empleado del video club ingresa los datos requeridos para asociar a una persona y el sistema registra un nuevo socio generando y asignándole un número único.

Registrar alquiler: el empleado del video club selecciona una película a alquilar por un socio en particular y el sistema registra el alquiler asignándole un número único y una fecha de vencimiento y actualiza el estado del ejemplar alquilado.

Actualizar datos socio: el encargado del video club selecciona un socio en particular y modifica uno o más datos del mismo a excepción del número de socio y el sistema registra la actualización.

Eliminar socio: el encargado del video club selecciona un socio en particular para eliminarlo y el sistema cambia el estado del socio de "activo" a "inactivo" y registra la actualización.

Generar listado de alquileres vencidos: el encargado del video club solicita un listado de alquileres vencidos a la fecha y el sistema genera y muestra por pantalla el listado incluyendo en el mismo los alquileres con fecha de vencimiento anterior a la fecha actual.

A modo de ejemplo se describirá en detalle el caso de uso *Registrar alquiler*:

Nombre del caso de uso: Registrar alquiler

Actor principal: Empleado del video club

Actor secundario: -

Precondición: Registrar socio, Registrar ejemplar

Postcodición: Alquiler registrado

Descripción general: el empleado del video club selecciona una película a alquilar por un socio en particular y el sistema registra el alquiler asignándole un número único y una fecha de vencimiento y actualiza el estado del ejemplar alquilado.

Interfaz tentativa:

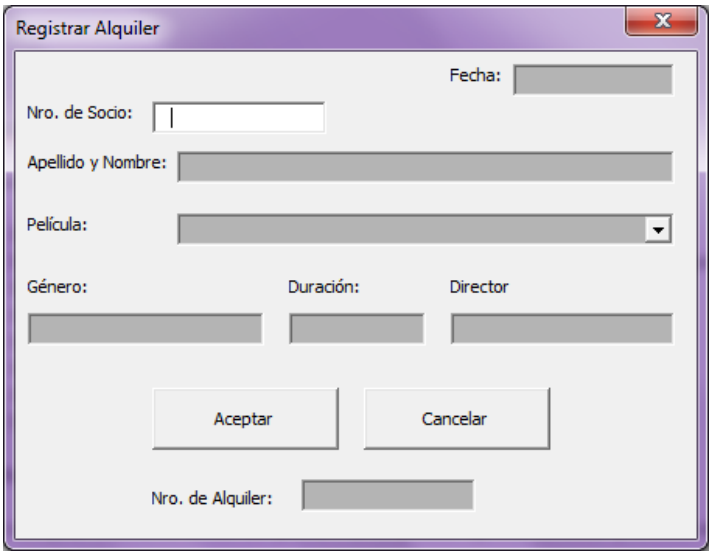

Figura 11: Interfaz del caso de uso registrar alquiler de película

Descripción detallada:

Flujo normal:

1. El empleado del video club selecciona Registrar alquiler
2. El sistema muestra la interfaz con la fecha y los nombres de las películas cargados
3. El empleado del video club ingresa el número de socio
4. El sistema muestra nombre y apellido del socio
5. El empleado del video club selecciona una película
6. El sistema muestra el género, duración y director de la película
7. El empleado del video club oprime aceptar
8. El sistema registra el alquiler y muestra el número de alquiler

Flujo alternativo:

*En cualquier momento antes del punto 7 el empleado oprime cancelar y finaliza el caso de uso.

4.1. Número de socio inexistente, el sistema informa la situación mostrando el cartel correspondiente y vuelve al paso 3.

6.1. No hay ejemplares disponibles para la película seleccionada, el sistema informa la situación mostrando el cartel correspondiente y vuelve al paso 5.

A continuación, se muestra una variante del caso de uso *Registrar alquiler* en la que un socio puede alquilar *más de una película* a la vez:

Nombre del caso de uso: Registrar alquiler

Actor principal: Empleado del video club

Actor secundario: -

Precondición: Registrar socio, Registrar ejemplar

Postcodición: Alquiler registrado

Descripción general: el empleado del video club selecciona las películas a alquilar por un socio en particular y el sistema registra el alquiler asignándole un número único y una fecha de vencimiento y actualiza el estado de los ejemplares alquilados.

Interfaz tentativa:

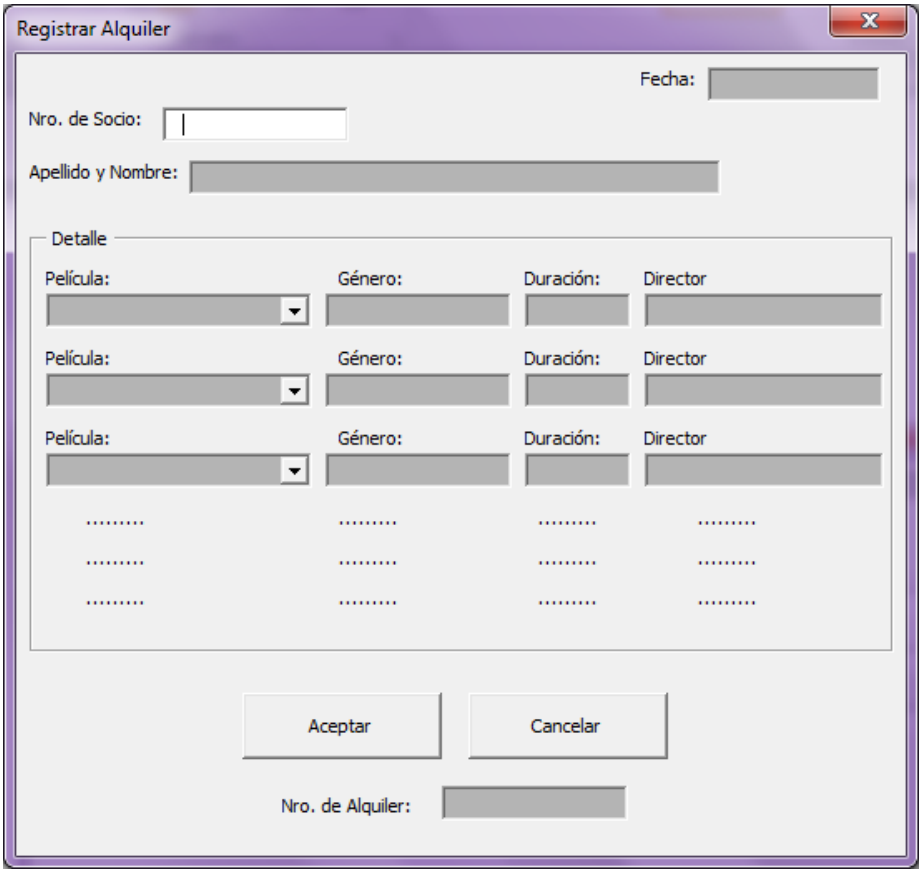

Figura 12: Interfaz del caso de uso registrar alquiler de más de una película

Descripción detallada:

Flujo normal:

1. El empleado del video club selecciona Registrar alquiler
2. El sistema muestra la interfaz con la fecha y los nombres de las películas cargados
3. El empleado del video club ingresa el número de socio
4. El sistema muestra nombre y apellido del socio
5. Para cada película:
    5.1. El empleado del video club selecciona una película
    5.2. El sistema muestra el género, duración y director de la película
6. El empleado del video club oprime aceptar
7. El sistema registra el alquiler y muestra el número de alquiler

Flujo alternativo:

*En cualquier momento antes del punto 6 el empleado oprime cancelar y finaliza el caso de uso.

4.1. Número de socio inexistente, el sistema informa la situación mostrando el cartel correspondiente y vuelve al paso 3.

5.2.1. No hay ejemplares disponibles para la película seleccionada, el sistema informa la situación mostrando el cartel correspondiente y vuelve al paso 5.1.

Sistema de reservas de habitaciones para una cadena hotelera

Una cadena de hoteles permite realizar la reserva de habitaciones vía web. La persona interesada primero dará sus datos personales para convertirse en cliente, luego reservará *un tipo de habitación* para un período determinado que posee una tarifa que depende del tipo de habitación y de la temporada (alta, media o baja). En el momento de la reserva no deberá abonar nada, pero si deberá hacerlo cuando confirme la misma, un mes antes del período elegido. Si la persona no confirma la reserva el sistema la dará de baja 20 días antes del periodo elegido. El hotel tiene un catálogo con los tipos de habitación con que cuenta y las tarifas de acuerdo a la temporada, que enviará dos veces por año a los clientes, los cuales podrán consultar el mismo.

Desarrollo del modelo de casos de uso

Actores: Visitante, Cliente, Empleado de reservas

Casos de uso para cada actor:

Visitante: Registrar cliente, Consultar catálogo.

Cliente: Ingresar al sistema, Consultar catálogo, Realizar reserva, Cancelar reserva, Confirmar reserva, Consultar datos reserva.

Empleado de reservas: Ingresar al sistema, Generar catálogo, Consultar catálogo, Actualizar datos catálogo, Actualizar datos cliente, Eliminar cliente, Eliminar reserva.

Diagrama de casos de uso[1]:

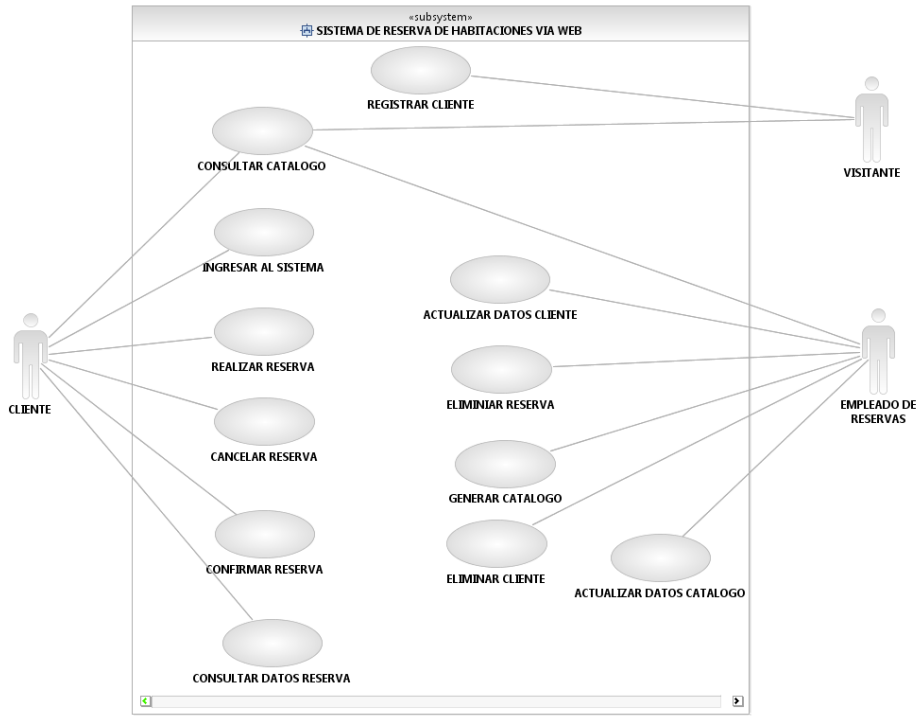

Figura 13: Diagrama de casos de uso del sistema de reservas de tipos de habitación

[1] Realizado con la herramienta IBM ® Rational® Software Architect™

Descripción de los casos de uso:

Registrar cliente: el visitante ingresa los datos requeridos para registrarse como cliente en el sitio web de la cadena hotelera y el sistema registra un nuevo cliente generando y asignándole un número único.

Consultar catálogo: el visitante, el cliente y/o el empleado de reservas selecciona un catálogo a consultar para un hotel en particular y el sistema muestra el catálogo seleccionado con los tipos de habitación y las tarifas vigentes según la temporada.

Ingresar al sistema: el cliente y/o el empleado de reservas ingresan los datos requeridos como usuario para iniciar sesión en el sistema y el sistema, de ser válidos los datos ingresados, habilita y muestra las funcionalidades correspondientes al perfil del usuario ingresado.

Realizar reserva: el cliente, para un hotel determinado, selecciona un tipo de habitación y un período y el sistema si encuentra disponibilidad registra la reserva asignándole un número de reserva único.

Cancelar reserva: el cliente selecciona una reserva que aún no ha confirmado para cancelarla y el sistema registra el cambio del estado de la reserva de “a confirmar” a “cancelada” y actualiza la disponibilidad del tipo de habitación reservado.

Confirmar reserva: el cliente selecciona una reserva que aún no ha confirmado para confirmarla e ingresa los datos requeridos para el pago de la misma y el sistema registra el pago y la actualización del estado de la reserva de “a confirmar” a “confirmada”.

Consultar datos reserva: el cliente selecciona una de sus reservas para consultar sus datos y el sistema muestra en pantalla los datos de la reserva seleccionada.

Generar catálogo: el empleado de reservas selecciona para cada hotel de la cadena los tipos de habitación y las tarifas correspondientes según la temporada para incluir en el catálogo y el sistema registra el catálogo con un período de vigencia determinado y un número de catálogo único.

Actualizar datos catálogo: el empleado de reservas selecciona un catálogo en particular y modifica uno o más datos del mismo a excepción del número de catálogo y el sistema registra la actualización.

Actualizar datos cliente: el empleado de reservas selecciona un cliente en particular y modifica uno o más datos del mismo a excepción del número de cliente y el sistema registra la actualización.

Eliminar cliente: el empleado de reservas selecciona un cliente en particular para eliminarlo y el sistema, si no encuentra reservas asociadas al mismo, cambia el estado del cliente de “activo” a “inactivo” y registra la actualización.

Eliminar reserva: el empleado de reservas selecciona una reserva que a la fecha no ha sido confirmada por el cliente y el sistema, de encontrarse la fecha actual dentro de los 20 días previos al período reservado, registra el cambio del estado de la reserva de “a confirmar” a “eliminada” y actualiza la disponibilidad del tipo de habitación reservado.

A modo de ejemplo se describirá en detalle el caso de uso *Realizar reserva* para un tipo de habitación:

Caso de uso: Realizar reserva

Actor principal: Cliente

Precondición: Registrar cliente, Registrar tipo de habitación, Registrar tarifa, Ingresar al sistema.

Postcondición: Reserva registrada

Descripción general: El cliente, para un hotel determinado, selecciona un tipo de habitación y un periodo y el sistema si encuentra disponibilidad registra la reserva asignándole un número de reserva único.

Interfaz tentativa:

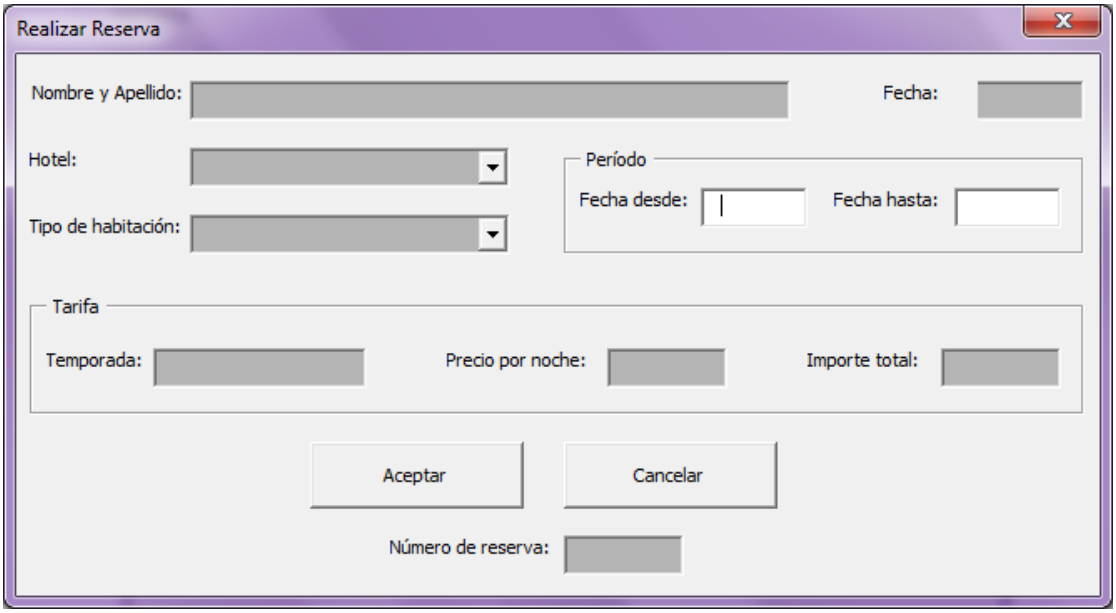

Figura 14: Interfaz del caso de uso registrar reserva de un tipo de habitación

Descripción detallada:

Flujo normal:

1. El cliente selecciona Realizar reserva
2. El sistema muestra la interfaz con la fecha, nombre y apellido del cliente y los hoteles cargados
3. El cliente selecciona un hotel
4. El sistema carga todos los tipos de habitación para el hotel seleccionado
5. El cliente selecciona un tipo de habitación e ingresa el periodo de tiempo

6. El sistema muestra la temporada, el precio por noche y el importe total

7. El cliente oprime aceptar

8. El sistema registra la reserva y muestra el número de reserva

Flujo alternativo:

*En cualquier momento antes del punto 7 el cliente oprime cancelar y finaliza el caso de uso.

6.1 No hay habitaciones del tipo seleccionado disponibles para el periodo seleccionado. El sistema muestra el mensaje correspondiente y vuelve al punto 5.

A continuación, se muestra una variante del caso de uso *Realizar reserva* en la que un cliente puede reservar *más de un tipo de habitación*:

Caso de uso: Realizar reserva

Actor principal: Cliente

Precondición: Registrar cliente, Registrar tipo de habitación, Registrar tarifa, Ingresar al sistema.

Postcondición: Reserva registrada

Descripción general: El cliente, para un hotel determinado y para cada tipo de habitación que desea reservar, selecciona un tipo de habitación e ingresa un periodo y el sistema si encuentra disponibilidad registra la reserva asignándole un número de reserva único.

Interfaz tentativa:

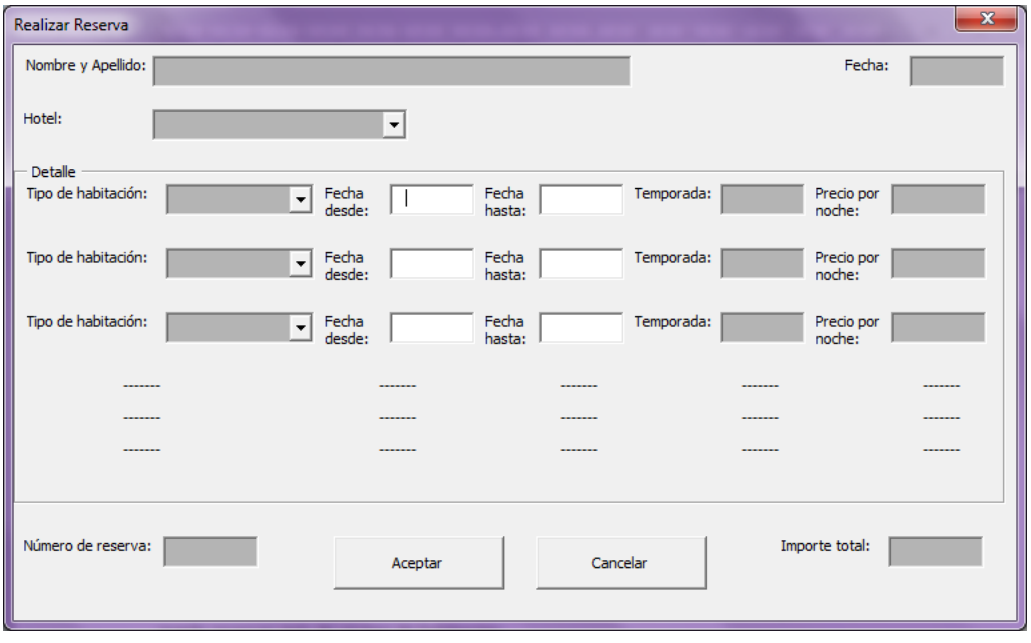

Figura 15: Interfaz del caso de uso registrar reserva de más de un tipo de habitación

Descripción detallada:

Flujo normal:

1. El cliente selecciona Realizar reserva

2. El sistema muestra la interfaz con la fecha, nombre y apellido del cliente y los hoteles cargados

3. El cliente selecciona un hotel

4. El sistema carga todos los tipos de habitación para el hotel seleccionado

5. Por cada tipo de habitación:

    5.1. El cliente selecciona un tipo de habitación e ingresa el periodo de tiempo

    5.2. El sistema muestra la temporada, el precio por noche y el importe total actualizado

6. El cliente oprime aceptar

7. El sistema registra la reserva y muestra el número de reserva

Flujo alternativo:

*En cualquier momento antes del punto 6 el cliente oprime cancelar y finaliza el caso de uso.

5.2.1 No hay habitaciones del tipo seleccionado disponibles para el periodo seleccionado. El sistema muestra el mensaje correspondiente y vuelve al punto 5.1.

Bibliografía:

Kendall & Kendall. (8va ed. 2011). *Análisis y Diseño de Sistemas.* Pearson - Pretince Hall.

Rumbaugh, J., Jacobson, I., & Booch, G. (1ra ed. 2000). *El Proceso Unificado de Desarrollo de Software.* Addison Wesley.

Rumbaugh, J., Jacobson, I., & Booch, G. (2da ed. 2006). *El Lenguaje Unificado de Modelado UML 2.0.* Addison Wesley.