UNIVERSIDAD DE ORIENTE.

NÚCLEO DE ANZOÁTEGUI.

ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS.

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS.

CÁTEDRA: DESARROLLO DE SOFTWARE (072-4712)



**Diagramas de UML:  
Diagrama de estructura compuesta  
Diagrama de casos de uso  
Diagrama de secuencia**

Autores:

|  |  |  |
| --- | --- | --- |
| Br. Francisco Noriega | Br. Yudimar Misel | Br. Marian Piñango |

Br. Osman Villegas

Barcelona, Mayo 2015

**Introducción**

El lenguaje unificado de diagrama o notación (UML) está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. Sirve para especificar, visualizar y documentar esquemas de sistemas de software orientado a objetos. UML no es un método de desarrollo, lo que significa que no sirve para determinar qué hacer en primer lugar o cómo diseñar el sistema, sino que simplemente le ayuda a visualizar el diseño y realizarlo de manera más accesible para otros de igual manera está controlado por el grupo de administración de objetos (OMG) y es el estándar de descripción de esquemas de software. Está diseñado para su uso con software orientado a objetos, y tiene un uso limitado en otro tipo de cuestiones de programación.

 Se compone de muchos elementos de esquematización que representan las diferentes partes de un sistema de software. Los elementos UML se utilizan para crear diagramas, que representa alguna parte o punto de vista del sistema

Los diagramas presentan diversas perspectivas de un sistema, a las cuales se les conoce como modelo. Recordemos que un modelo es una representación simplificada de la realidad; el modelo UML describe lo que supuestamente hará un sistema, pero no dice cómo implementar dicho sistema.

**Diagramas UML**

* **Diagrama de estructura compuesta**

Tipo de diagrama de estructura estática en el lenguaje de modelado unificado (uml), que muestra la estructura interna de una [clase](http://es.wikipedia.org/wiki/Clase_(inform%C3%A1tica)) y las colaboraciones que esta estructura hace posibles. Esto puede incluir partes internas, puertas mediante las cuales, las partes interactúan con cada una de las otras o mediante las cuales, instancias de la clase interactúan con las partes y con el mundo exterior, y conectores entre partes o puertas. Una estructura compuesta es un conjunto de elementos interconectados que colaboran en tiempo de ejecución para lograr algún propósito. Cada elemento tiene algún rol definido en la colaboración.

**Elementos de estructura compuesta**

**Parte**

Se muestra con un rectángulo e indica los objetos que conforman el objeto principal.

Las partes son instancias en tiempo de ejecución de clases o interfaces.

**Puerta**

Una *puerta* es un punto de interacción que puede ser usado para conectar clasificadores estructurados con sus partes y con el ambiente. Las puertas pueden opcionalmente especificar los servicios que proveen y los servicios que requieren de otras partes del sistema.

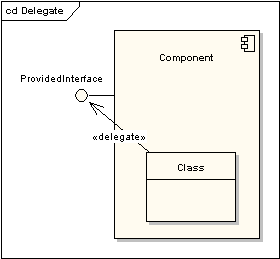
Las puertas pueden ya sea delegar los requerimientos recibidos a partes internas, o pueden entregarlos directamente para el comportamiento del clasificador estructurado en el que la puerta está contenido. Las puertas públicas que son visibles en el ambiente son mostradas sobre el borde (límite o frontera), mientras que las puertas protegidas que no son visibles en el ambiente son mostradas dentro de la frontera (borde o límite).

**Conector**

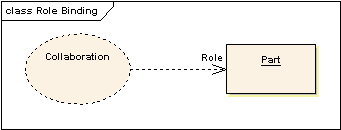
Un conector une dos o más entidades, permitiéndoles interactuar en tiempo de ejecución. Un conector es representado por una línea que une una combinación de partes, puertas y [clasificadores](http://es.wikipedia.org/wiki/Clasificador) estructurados.

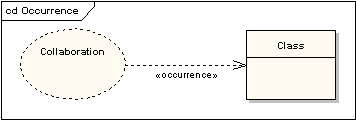
**Tipos de conector**

* Delegar  
  Un conector delegar se usa para definir los trabajos internos de los puertos e interfaces externas del componente. Un conector delegar se muestra como una flecha con un estereotipo «delegar». Esto conecta un contrato externo de un componente como se muestra por sus puertos a la realización interna del comportamiento de la parte del componente.



* **Enlace de Roles**

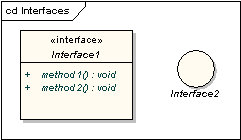
Un conector enlace de roles se dibuja desde una colaboración a un clasificador que completa el rol. Esto se muestra como una línea de trazos con una punta de flecha y el estereotipo «enlace de roles».

* Representa  
  Un conector representa se puede dibujar desde una colaboración a un clasificador para mostrar que una colaboración se usa en el clasificador. Se muestra como una línea de trazos con una punta de flecha y el estereotipo «representa».
* Ocurrencia  
  Un conector ocurrencia se puede dibujar desde una colaboración a un clasificador para mostrar que la colaboración representa (sic) el clasificador. Esto se muestra como una línea de trazos y el estereotipo «ocurrencia».

Una *colaboración* es generalmente más abstracta que un clasificador estructurado. Ésta es mostrada como un óvalo sin relleno conteniendo los roles que las instancias pueden jugar en la colaboración.

**Clasificador estructurado**

Un *Clasificador Estructurado* representa una clase, frecuentemente una clase abstracta, cuyo comportamiento puede ser completa o parcialmente descrito mediante interacciones entre partes.

Interfaces  
Una interfaz es similar a una clase pero con un número de restricciones. Todas las operaciones de la interfaz son públicas y abstractas, y no proveen ninguna implementación predeterminada. Todos los atributos de la interfaz deben ser constantes. Sin embargo, mientras que una clase puede solo heredar de una sola super-clase, puede implementar interfaces múltiples.

Una interfaz, cuando está sola en un diagrama, se muestra como un rectángulo del elemento clase con  la clave «interfaz» y con su nombre en itálica para denotar que es abstracto, o se muestra como un circulo.

* **Casos de Uso**

Antes de entrar en detalles de que son los casos de uso podemos decir que el análisis de cómo se puede utilizar una casa es un ejemplo de un estudio basado en casos de uso ya que se consideran las diferentes formas en que se utilizara la casa, y estos casos de uso guían la arquitectura de esta. En muchas casas tendrán las mismas categorías de casos de uso ya que estas se utilizan para comer, dormir, criar a los niños etc. Pero cada casa también tendrá sus propios casos de usos que podrán ser especiales o variaciones de los básicos.

Un factor clave al definir casos de usos es que no se especifica cómo se implementan sino más bien estos especifican el comportamiento de cómo se verá desde afuera, pero no imponen como se llevara a cabo internamente ese comportamiento. En cierto punto lo más importante es que permiten que los usuarios finales y los expertos se comuniquen con los desarrolladores sin quedarse atascados en los detalles.

A nivel del sistema un caso de uso describe un conjunto de secuencias, donde cada secuencia representa la interacción de los elementos externos al sistema como son los actores o los usuarios con el propio sistema, donde este comportamiento o interacción se utilizan durante la captura de requisitos y el análisis para visualizar, especificar, construir y documentar el comportamiento esperado por el sistema.

Un caso de uso involucra la interacción de los actores y el sistema u otros sujetos.

Un actor representa un conjunto coherente de roles que juegan los usuarios de los casos de uso al interactuar con estos en donde estos pueden ser personas o pueden ser sistemas automáticos.

Un caso de uso puede tener variantes.

Se pueden encontrar casos de usos que son versiones especializadas de otros casos de uso o que pueden ser incluidos como partes de otros o que extienden el comportamiento de otros más básicos.

Un caso de uso realiza cierto trabajo cuyo efecto es tangible

Un caso de uso produce algo de valor para algún actor, como puede ser el cálculo de un resultado, la generación de un nuevo objeto o el cambio del estado de otro objeto

Los casos de uso se pueden aplicar al sistema completo

Esto casos de uso no solo representan el comportamiento esperado de estos elementos, sino que también pueden utilizarse como la base para establecer casos de prueba para los elementos mientras evolucionan durante el desarrollo del sistema

Los casos de usos aplicados a los subsistemas son una fuente excelente de pruebas de regresión en otro sentido es que cuando son aplicados al sistema completo son una fuente excelente de pruebas del sistema y de integración

**Caso de uso**

Un caso de uso es una descripción de un conjunto de secuencias de acciones, incluyendo variantes que ejecuta un sistema para producir un resultado observable de valor para un actor y este se representa gráficamente como una elipse

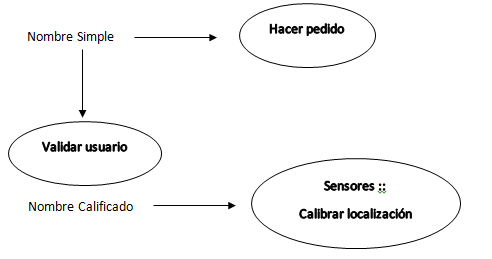
**Sujeto**

El sujeto es una clase descrita por un conjunto de casos de uso. Normalmente la clase es un sistema o un subsistema. El caso de uso representa aspectos del comportamiento de la clase y los actores aspecto de otras clases que interactúan con el sujeto. Uniendo todo los casos de uso describen el comportamiento completo del sujeto.

**Nombres**

Cada caso de uso debe tener un nombre que lo distinga de otros casos de uso. Existen dos tipos de nombres:

* Nombre simple: Es una cadena de texto.
* Nombre Calificado: este consta del nombre del caso de uso precedido del nombre del paquete en el que se encuentra



Nota: El nombre puede constar de texto con cualquier número de letras, números y la mayoría de los signos de puntuación (excepto signos como los dos puntos “ :: “ que son utilizados para separar el nombre de un caso de uso del nombre del paquete que lo contiene) y pueden extenderse a lo lardo de varias líneas.

**Casos de uso y actores**

Un actor representa un conjunto coherente de roles que los usuarios de los casos de usos representan al interactuar con estos. Normalmente estos actores representan un rol que es desempeñado por una persona, un dispositivo hardware o incluso otro sistema al interactuar con nuestro sistema.

En un sistema en ejecución estos no han de existir como entidades separadas. Un objeto puede representar el papel de varios actores.

Por ejemplo:

Si una persona trabaja para un banco, está desempeña rol de empleado y además puede ejercer el rol de cliente si posee cuentas personales en ese banco

Actor Actor

Generalización

Empleado Cliente

**Casos de uso y flujo de eventos**

Un caso de uso describe que hace un sistema pero no especifica cómo lo hace. Es importante tener clara la separación de intereses entre las vistas externar e internas.

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 esté se debe incluir cómo y cuándo empieza y acaba el caso de uso, cuando interactúa con los actores y que objetos se intercambian, el flujo básico y los flujos alternativos del comportamiento.

Por ejemplo

Con un cajero automático se puede describir el caso de uso VALIDA\_USUARIO de la siguiente manera:

* **Flujo de eventos principal:** 
  + El caso de uso comienza cuando el sistema pide al CLIENTE un número de identificación personal (PIN).
  + El Cliente puede introducir su PIN a través del teclado.
  + El Cliente acepta la entrada pulsando el botón ENTER. El sistema comprueba entonces este PIN para ver si es válido. Si el PIN es válido, el sistema acepta la entrada y así culmina el caso de uso.
* **Flujos de eventos excepcionales:**
  + El Cliente puede cancelar una transacción en cualquier momento pulsando el botón CANCELAR y de esta forma reinicia el caso de uso.
  + El Cliente puede borrar un PIN en cualquier momento antes de introducirlo y volver a teclear un nuevo PIN.
  + Si el Cliente introduce un PIN invalido, el caso de uso vuelve a empezar. Si esto ocurre tres veces en una sesión el sistema cancela la transacción lo que impide que el Cliente utilice el cajero durante 60 segundos.

**Nota:**

El flujo de eventos de un caso de uso se puede especificar de muchas formas como el texto estructurado informal, el formal, máquinas de estados, diagramas de actividad y pseudocódigo.

**Casos de uso y escenarios**

El flujo de eventos de un caso de uso primero se describe mediante texto luego conforme se mejora la comprensión de los requisitos del sistema estos flujos se pueden especificar gráficamente por medio de diagramas de interacción pero normalmente se usa un diagrama de secuencia para especificar el flujo principal de un caso de uso y se usan variaciones de ese diagrama para especificar los flujos excepcionales de esté.

Es conveniente separar el flujo principal de los flujos alternativos, porque un caso de uso describe un conjunto de secuencias mas no una única secuencia, y sería imposible expresar todos los detalles de un aso de uso no trivial en una única secuencia.

**Por ejemplo**

Un sistema de recursos humanos podría aparecer el caso de uso CONTRATAR EMPLEADO donde esta función del sistema puede tener múltiples variantes.

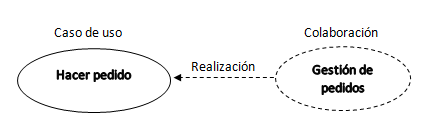
* Podría contratarse a una persona de otra empresa.
* Podría trasladarse una persona de un departamento a otro.
* Podría contratarse a un extranjero.

Cada una de estas variantes se puede expresar en una secuencia diferente, donde cada secuencia representa un posible flujo a través de todas las variantes. Cada secuencia se denomina escenario.

Un escenario es una secuencia específica de acciones que ilustra un comportamiento. Los escenarios son a los casos de uso lo que las instancias a las clases, en otras palabras, un escenario es básicamente una instancia de un caso de uso.

**Caso de uso y colaboraciones**

Un caso de uso captura el comportamiento esperado del sistema que se está desarrollando, sin tener que especificar cómo se implementa ese comportamiento. Esta separación es importante porque el análisis de un sistema no debería estar influenciado por cuestiones de implementación. No obstante, un caso de uso debe implementarse al fin y al cabo, y esto se hace creando una sociedad de clases y otros elementos que colaboraran para llevar a cabo el comportamiento del caso de uso. Esta sociedad de elementos, incluyendo tanto su estructura estática como la dinámica, se modela en UML como una colaboración.



Esta representación no muestra la realización de un caso de uso que puede especificarse explícitamente mediante una colaboración. Pero, aunque la mayoría de las veces un caso de uso es realizado exactamente por una colaboración, no será necesario mostrar explícitamente esta relación.

**Organización de los casos de uso**

Los casos de uso pueden organizarse agrupándolos en paquetes, de la misma manera que se organizan las clases.

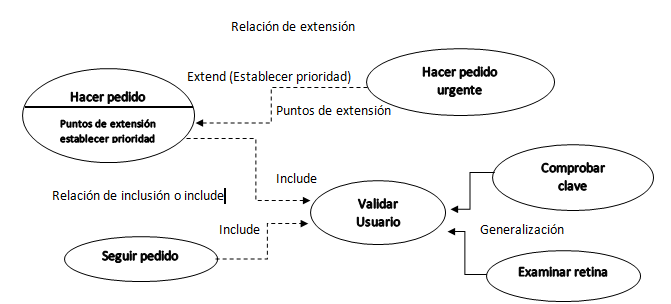
Los casos de usos también pueden organizarse especificando relaciones de generalizaciones, inclusión (INCLUDE) y extensión (EXTENDS) entre ellos. Estas relaciones se utilizan para factorizar el comportamiento común y para factorizar variantes.

La generalización entre casos de uso es como la generalización entre clases. Aquí significa que el caso de uso hijo hereda el comportamiento y el significado del caso de uso padre.

* El hijo puede añadir o redefinir el comportamiento del padre.
* El hijo puede ser colocado en cualquier lugar donde aparezca el padre.

**Por ejemplo**

Un sistema bancario puede tenerse el caso de uso VALIDAR USUARIO, responsable de verificar la identidad del usuario. Además, podría haber dos hijos especializados de este caso de usos COMPROBAR CLAVE y EXAMINAR RETINA los cuales se comportarían como VALIDAR USUARIO y podrían utilizarse donde quiera que apareciera VALIDAR USUARIO aunque los dos tengan sus propios comportamientos.



**Inclusión (INCLUDE)**

Una relación de inclusión entre casos de uso significa que un caso de uso base incorpora explícitamente el comportamiento de otro caso de uso en el lugar especificado en el caso base.

El caso de uso incluido nunca se encuentra aislado, sino que es instanciado solo como parte de algún caso de uso más amplio que lo incluye. La inclusión puede verso como que el caso de uso base toma el comportamiento del caso de uso proveedor. Esta relación de inclusión se usa para evitar describir el mismo flujo de eventos repetidas veces, poniendo el comportamiento común en un caso de uso aparte.

La relación de inclusión es básicamente un ejemplo de delegación porque se toma un conjunto de responsabilidades del sistema y se capturan en un sitio y a continuación se permite que otras partes del sistema incluyan la nueva agregación de responsabilidades siempre y cuando que se necesite usar esa funcionalidad

Una relación de inclusión se representa con INCLUDE. Para especificar la posición en un flujo de eventos en la cual el caso de uso base incluye el comportamiento de otro caso de uso, simplemente se debe escribir INCLUDE seguido del nombre del caso de uso que se quiere incluir, como en el flujo de SEGUIR PEDIDO.

**Extensión (EXTENDS)**

Una relación de extensión entre casos de uso significa que un caso de uso base incorpora implícitamente el comportamiento de otro caso de uso en el lugar especificado indirectamente por el caso de uso que extiende al base. El caso de uso base puede estar aislado pero en algunas condiciones su comportamiento puede extenderse con el comportamiento de otro caso de uso. Este caso de uso base puede extenderse solo en ciertos punto llamados puntos de extensión.

Una relación de extensión se utiliza de varias maneras:

* Se puede utilizar para modelar la parte de un caso de uso que el usuario puede ver como comportamiento opcional del sistema ya que de esta forma se separa el comportamiento opcional del obligatorio.
* Se puede utilizar para modelar un subflujo separado que se ejecuta solo en ciertas condiciones.
* Se puede utilizar para modelar varios flujos que se pueden insertar en un punto dado, controlados por la interacción explicita con un actor.
* Se puede usar para distinguir las partes opcionales de un sistema que se va implementar

Una relación de extensión se representa como EXTEND y los puntos de extensión del caso de uso se pueden listar en un comportamiento extra.

**Otras características (casos de uso clasificadores)**

Los casos de uso también son clasificadores de forma que pueden tener operaciones y atributos que se pueden representar igual que en las clases. Como clasificadores que son también se pueden asociar máquinas de estados a los casos de uso en donde estas se pueden usar como otra forma de describir el comportamiento representado por un caso de uso.

**Técnica de modelado**

Modelado del comportamiento de un elemento

La mayoría de las veces, los casos de uso se utilizan para el modelado del comportamiento de un elemento, ya sea un sistema completo, un subsistema o una clase. Cuando se modela el comportamiento de estos elementos, es importante centrarse en lo que hace el elemento, no en cómo lo hace.

Es importante aplicar de esta forma los casos de uso a los elementos por tres razones:

* El modelado del comportamiento de un elemento mediante casos de uso permite a los expertos del dominio especificar su vista externa del sistema a nivel suficientemente para que los desarrolladores construyan su vista interna. Ya que los casos de uso proporcionan un foro en el que puedan intercambiar opiniones los expertos, los usuarios finales y los desarrolladores.
* Los casos de uso proporcionan a los desarrolladores una forma de abordar y comprender un elemento. Un sistema, un subsistema o una clase pueden ser muy complejos y estar repletos de operaciones y otras partes. Cuando se especifican los casos de uso de un elemento se ayuda a los usuarios de ese elemento lo puedan entender. En ausencia de estos los usuarios tendrían que descubrir cómo usar el elemento.
* Los casos de uso sirven de base para probar cada elemento conforme evoluciona durante el desarrollo.

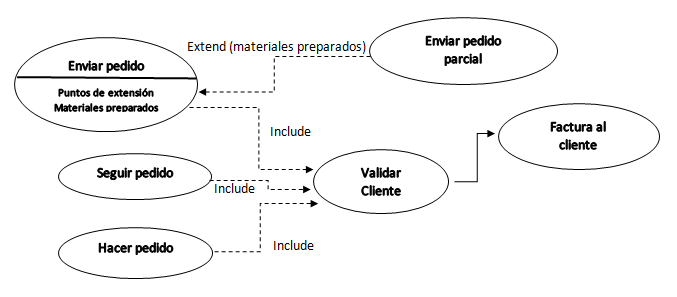
Estos casos de uso no solo sirven como punto de partida, sino que cada vez que se añade un elemento hay que reconsiderar la implementación para asegurarse de que se elemento es flexible al cambio y si no lo es la arquitectura debe reorganizarse del modo adecuado

Para modelar el comportamiento de un elemento:

* Hay que identificar los actores que interactúan con el elemento.
* Hay que organizar los actores identificando tanto los roles más generales como los especializados.
* Hay que considerar las formas más importantes que tiene cada actor de interactuar con el elemento
* Hay que considerar las formas excepcionales en las que cada actor puede interactuar con el elemento
* Hay que organizar estos comportamientos como casos de uso, utilizando las relaciones de inclusión y extensión para factorizar el comportamiento común y distinguir el comportamiento excepcional

**Por ejemplo**

Un sistema de ventas interactuara con clientes, que efectuaran pedido y querrán llevar un seguimiento, a su vez, el sistema enviara pedidos y facturas al cliente.



* **Diagramas de casos de Uso**

Los diagramas de caso de uso son uno de los tipos de diagramas que se utilizan para modelar los aspectos dinámicos de un sistema. Los diagramas de casos de uso son importantes para modelar el comportamiento de un sistema, subsistema o una clase. Cada uno muestrea un conjunto de casos de usos, actores y sus relaciones.

Éste es un tipo especial de diagrama y comparte las propiedades comunes al resto de los diagramas (un nombre y un contenido gráfico que es una proyección del modelo).

Los diagramas de casos de uso se emplean para modelar la vista de casos de uso de un sistema. La mayoría de las veces, esto implica modelar el contexto del sistema subsistema o clase, o el modelado de los requisitos del comportamiento de esos elementos.

Éstos son importantes para visualizar, especificar y documentar el comportamiento de un elemento. También facilitan que los sistemas, subsistemas y clases sean abordables y comprensibles, al presentar una vista externa de cómo pueden utilizarse estos elementos en un contexto dado. Los diagramas de casos de uso también son importantes para probar sistemas ejecutables a través de ingeniería inversa.

**Contenido**

Normalmente, un diagrama de caso de uso contiene:

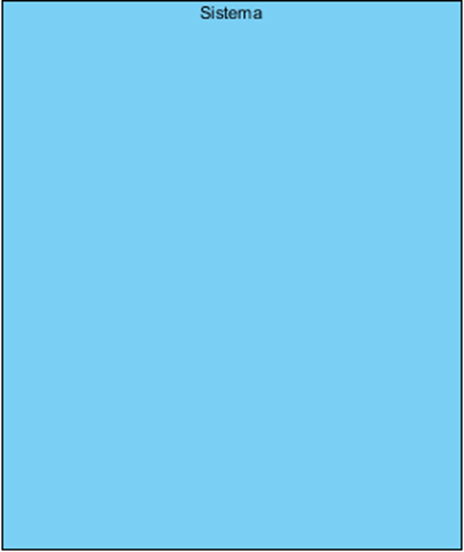
* Sujetos.
* Casos de uso.
* Actores.
* Relaciones de dependencia, generalización y asociación.

Al igual que los otros diagramas, los diagramas de casos de uso pueden contener notas y restricciones.

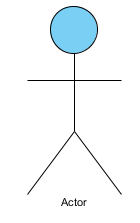
Los diagramas de casos de uso también pueden contener paquetes, que se emplean para agrupar elementos del modelo en partes mayores. De vez en cuando, se pueden incluir instancias de casos de uso en los diagramas, especialmente cuando se quiera visualizar un sistema específico en ejecución.

**Notación**

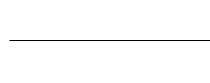
* **Sujeto:**  se representa con un rectángulo



*  **Casos de uso:** se representa a través de una elipse
* **Los actores:** Se muestran como monigotes fuera del rectángulo, con el nombre debajo



* **Las líneas:** conectan los íconos de los actores con las elipses de los casos de uso con los que se comunican.



* **Las relaciones**  entre los casos de uso (como la extensión y la inclusión) se dibujan dentro del rectángulo.



**Aplicación de los diagramas de casos de uso**

Los diagramas de casos de uso se emplean para modelar la vista de los casos de uso de un sujeto como un sistema. Esta vista abarca principalmente el comportamiento externo del sujeto.

Cuando se modela la vista de casos de uso estática de un sujeto, normalmente se emplean los diagramas de caso de uso de una de las dos formas siguientes:

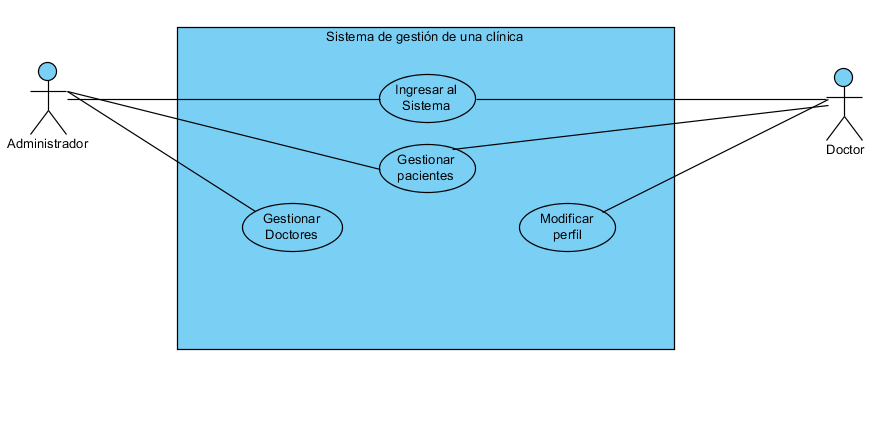
1. **Modelar el contexto de un sujeto**

Modelar el contexto de un sujeto implica dibujar una línea alrededor de todo el sistema e indicar qué actores queda fuera del sistema e interactúan con él. Aquí se emplean los diagramas de casos de uso para especificar los actores y el significado de sus roles.

Para modelar el contexto del sistema:

* Hay que identificar las fronteras del sistema decidiendo los comportamientos que formarían parte de él y cuáles serán ejecutados por entidades externas. Esto define el sujeto.
* Hay que identificar los actores en torno al sistema, considerando qué grupos requieren ayuda del sistema para llevar a cabo sus tareas; qué grupos son necesarios para ejecutar las funciones del sistema; qué grupos interactúan con el hardware externo o con otros sistemas software; y qué grupos realizan funciones secundarias de administración y mantenimiento.
* Hay que organizar los actores similares en jerarquías de generalización/ especificación.
* Hay que proporcionar un estereotipo para cada uno de estos actores, si así se ayuda a entender el sistema. Hay que modelar esos casos de uso, actores y relaciones en un diagrama de casos de uso.

A continuación se muestra el modelado del contexto de un sistema que permitirá gestionar pacientes y doctores de una clínica.



Modelado del contexto del sistema

1. **Modelar los requisitos de un sujeto**

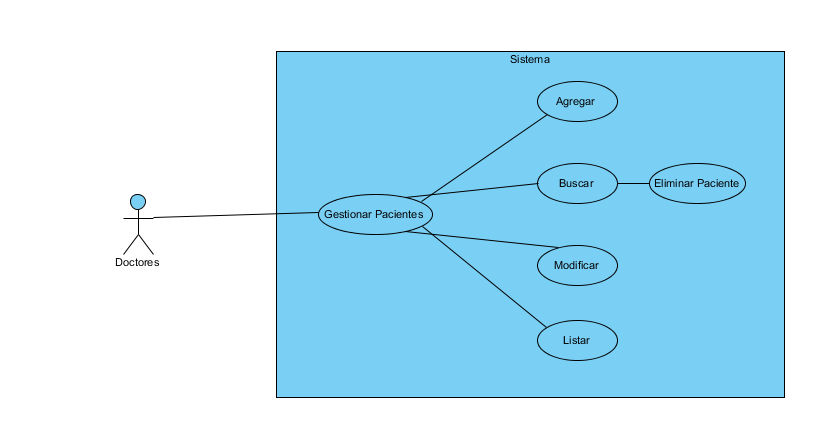
El modelado de requisitos de un sujeto implica especificar qué debería hacer ese sujeto (desde el punto de vista externo), independientemente de cómo lo haga. Aquí se emplearán los diagramas de casos de uso para especificar el comportamiento deseado del sistema. De esta forma, un diagrama de casos de uso para especificar el comportamiento deseado del sistema. De esta forma, un diagrama de casos de uso permite ver el objeto entero como una caja negra; se puede ver qué hay fuera del sujeto y cómo reacciona a los elementos externos, pero no se puede ver cómo funciona por dentro.

Para modelar los requisitos de un sistema:

* Hay que establecer el contexto del sistema, identificando los actores a su alrededor.
* Hay que considerar el comportamiento que cada actor espera del sistema o requiere que éste le proporcione.
* Hay que nombrar esos comportamientos comunes como casos de uso.
* Hay que factorizar el comportamiento común en nuevos casos de uso que puedan ser utilizados por otros; hay que factorizar el comportamiento variante en nuevos casos de uso que extiendan los flujos principales.

El ejemplo que se muestra a continuación extiende del diagrama de caso de uso anterior partiendo del caso de uso Ingresar al Sistema mostrándose así todas las acciones que se ejecutan desde que el actor ingresa al sistema, siendo éste a la vez un subsistema.

Modelado de los requisitos del sistema



* **Diagramas de secuencia**

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso. Mientras que el [diagrama de casos de uso](http://es.wikipedia.org/wiki/Diagrama_de_casos_de_uso) permite el modelado de una vista *business* del escenario, el diagrama de secuencia contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario y mensajes intercambiados entre los objetos.

Típicamente se examina la descripción de un [caso de uso](http://es.wikipedia.org/wiki/Caso_de_uso) para determinar qué objetos son necesarios para la implementación del escenario. Si se dispone de la descripción de cada [caso de uso](http://es.wikipedia.org/wiki/Caso_de_uso) como una secuencia de varios pasos, entonces se puede "caminar sobre" esos pasos para descubrir qué objetos son necesarios para que se puedan seguir los pasos. Un diagrama de secuencia muestra los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como flechas horizontales.

**Tipos de mensajes**

Existen dos tipos de mensajes: sincrónicos y asincrónicos. Los mensajes sincrónicos se corresponden con llamadas a métodos del objeto que recibe el mensaje. El objeto que envía el mensaje queda bloqueado hasta que termina la llamada. Este tipo de mensajes se representan con flechas con la cabeza llena. Los mensajes asincrónicos terminan inmediatamente, y crean un nuevo hilo de ejecución dentro de la secuencia. Se representan con flechas con la cabeza abierta.

También se representa la respuesta a un mensaje con una flecha discontinua.

**Estructura**

**•** El diagrama de secuencias consta de objetos que se representan del modo usual: rectángulos con nombre (subrayado), mensajes entre los objetos representados por líneas continuas con una punta de flecha y el tiempo representado como una progresión vertical.

• Los objetos se colocan cerca de la parte superior del diagrama de izquierda a derecha y se acomodan de manera que simplifiquen el diagrama.

• La extensión que está debajo (y en forma descendente) de cada objeto será una línea discontinua conocida como la línea de vida de un objeto.

• Junto con la línea de vida de un objeto se encuentra un pequeño rectángulo conocido como activación, el cual representa la ejecución de una operación que realiza el objeto. La longitud del rectángulo se interpreta como la duración de la activación.

• Un mensaje que va de un objeto a otro pasa de la línea de vida de un objeto a la de otro. Un objeto puede enviarse un objeto a sí mismo (es decir, de su línea de vida a su propia línea de vida).

• Un mensaje puede ser simple, síncrono o asíncrono.

• Un mensaje simple es la transferencia del control de un objeto a otro.

• En el diagrama de secuencias, los símbolos del mensaje varían. Por ejemplo, la punta de la flecha de un mensaje simple está compuesta por dos líneas, la punta de flecha de un mensaje síncrono es un triángulo relleno, y la de uno asíncrono solo tiene una sola línea.

El diagrama representa al tiempo en dirección vertical. El tiempo se inicia en la parte superior y avanza hacia la parte inferior. Un mensaje que esté más cerca de la parte superior ocurrirá antes que uno que esté cerca la parte inferior.

Con ello el diagrama de secuencias tiene dos dimensiones. La dimensión horizontal es la disposición de los objetos, y la dimensión vertical muestra el paso del tiempo.

En ocasiones un objeto posee una operación que se invoca a sí misma. A esto se le conoce como recursividad, y es una característica fundamental de varios lenguajes de programación. Por ejemplo, supongamos que una calculadora forma parte de los objetos de nuestro sistema y que una de sus operaciones sea el cálculo de intereses. Para calcular el interés compuesto para un periodo que incluya otros periodos, la operación cálculo de intereses del objeto tendrá que invocarse a si misma varias veces. Para representar esto en UML, dibujaremos una flecha de mensaje fuera de la activación, y un pequeño rectángulo encima de la activación.

**Conclusión**

UML resuelve de forma bastante satisfactoria un viejo problema del desarrollo de software como es su modelado gráfico. Además, se ha llegado a una solución unificada basada en lo mejor que había hasta el momento, lo cual lo hace todavía más excepcional.

El diagrama de casos de usos representa gráficamente los casos de uso que tiene un sistema. Se define un caso de uso como cada interacción supuesta con el sistema a desarrollar, donde se representan los requisitos funcionales. Es decir, se está diciendo lo que tiene que hacer un sistema y cómo. En la figura 3 se muestra un ejemplo de casos de uso, donde se muestran tres actores (los clientes, los taquilleros y los jefes de taquilla) y las operaciones que pueden realizar (sus roles).

En el diagrama de secuencia se muestra la interacción de los objetos que componen un sistema de forma temporal. Siguiendo el ejemplo de venta de entradas, la figura 5 muestra la interacción de crear una nueva sala para un espectáculo.

El resto de diagramas muestran distintos aspectos del sistema a modelar. Para modelar el comportamiento dinámico del sistema están los de interacción, colaboración, estados y actividades. Los diagramas de componentes y despliegue están enfocados a la implementación del sistema.

**Bibliografía**

* Martin Fowler, Kendall Sccott (1999), "UML Gota a Gota".
* Perdita Stevens, Rob Pooley. Addison Wesley. (2002). Utilización de UML en Ingeniería del Software con Objetos y Componentes.
* Grady Booch, James Rumbaugh (2006) “El Lenguaje Unificado de Modelado, Guía de Usuario”.
* Grady Booch, James Rumbaugh, Ivar Jacobson (2007) “El Lenguaje Unificado de Modelado, Manual de Referencia”.