DIAGRAMAS DE CASOS DE USO

© *Luis Pilo Aceituno* lpilo@educarex.es

Los diagramas de ***casos de uso*** documentan el comportamiento de un sistema desde el punto de vista del usuario.

- ***Los casos de uso determinan los requisitos funcionales del sistema, es decir, representan las funciones que el sistema puede realizar.***

Los casos de uso son especialmente útiles para comunicarnos con el cliente y hacerle entender la funcionalidad del sistema que desarrollaremos.

Los elementos básicos que componen un diagrama de casos de uso son:

- Actores
- Caso de uso
- Asociaciones
  - Include
  - Extended
  - Generación.
- Escenario.

ACTORES

Un actor representa una entidad del mundo real que da o recibe directamente información del sistema

Los actores representan un tipo de usuario del sistema. Se entiende como usuario cualquier cosa externa que interactúa con el sistema. No tiene por qué ser un ser humano, puede ser otro sistema informático o unidades organizativas o empresas

Los actores puede ser:

1. Usuarios que interactúan directamente con el software (sistema informático)
2. Dispositivos que no sean simples periféricos, que intercambian información con el sistema.
3. Otro software con el que comparte ficheros
4. El simple paso del tiempo en el caso de los procesos que se mandan automáticamente a una hora determinada.

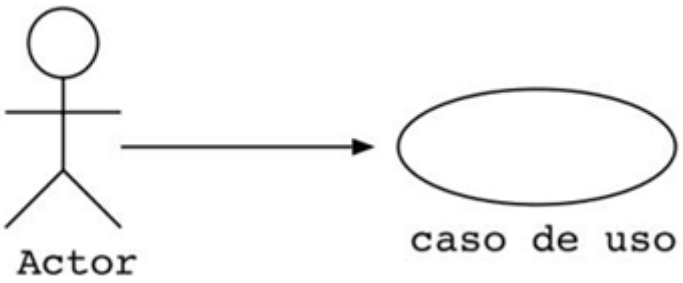
- Hay que independizar los actores de la forma en que se interactúa con el sistema, por ejemplo, un teclado no es un actor sino un medio para introducir información en el sistema
- Además puede ser útil mantener una lista de los usuarios reales para cada actor.

Un actor representa un rol que alguien puede estar jugando, no un individuo particular por tanto puede haber personas que puedan estar usando el sistema de formas diferentes en diferentes ocasiones

---

EJEMPLO: Si una persona trabaja para un banco puede ser "responsablePrestamos" y si además tiene sus cuentas en ese banco ser un "cliente"

Un actor se representa con un dibujos simplificados de personas ' y el caso de uso con una elipse.



CASOS DE USO

Un caso de uso especifica una función que debe realizar el sistema pero no como hacerlo. Se representa mediante un óvalo

- ***Un caso de uso es una secuencia de interacciones entre un sistema y alguien o algo que usa alguno de sus servicios. Un caso de uso es iniciado por un actor.***
- ***El nombre de un caso de uso se expresa por un verbo en gerundio. El nombre del caso de uso siempre se expresa desde el punto de vista del actor y no desde el punto de vista del sistema***

Los casos de uso tienen las siguientes características.

1. Están expresados desde el punto de vista del actor
2. Se documentan con texto informal
3. Describen tanto lo que hace el actor como lo que hace el sistema cuando interactúa con él.
4. Son iniciados por un actor
5. Están acotados al uso de una determinada funcionalidad, claramente diferenciada, del sistema.

La pregunta ¿Qué es una “funcionalidad claramente diferenciada” es difícil de responder.. Por ejemplo. ¿ingresar pedidos es un caso de uso y autorizarlos ? Cancelar los pedidos, es otro caso de uso, o es parte del caso de uso referido al ingreso de pedidos?. La respuesta es que todos son casos de uso distintos.. Si en la programación los criterios para dividir la funcionalidad en programas suelen ser difusos, los criterios para dividir la funcionalidad de un sistema en casos de uso son aún más difusos. y por eso se hace importante utilizar el sentido común al tomar estas decisiones.

La regla general es: una función del sistema es un caso de uso si se debe indicar explícitamente al sistema que uno quiere acceder a esa función.

Por ejemplo, si uno quiere dar de alta un pedido deberá acceder a la funcionalidad de alta pedidos del sistema. Sin embargo, si uno quiere dar de alta un campo del pedido, no debe indicar al sistema que quiere acceder a esa función. Dar de alta un campo de un pedido es una función que forma parte de un caso de uso mayor: dar de alta un pedido

Esta regla, no obstante, no debe seguirse al pie de la letra. Por ejemplo supongamos que uno quiere especificar un sistema en el cual los usuarios pueden ver un pedido y tiene funciones disponibles para ver el siguiente pedido, el anterior, el último y el primero. El actor debe indicar al sistema que quiere acceder a cada una de esas funciones y según nuestra regla serán todos casos de uso distintos. Sin embargo es más práctico definir un único caso de uso ***navegando pedidos*** que especificarlos todos como casos de uso distintos.

Cuando pensamos en el grado de detalle de la división de los casos de uso también es útil imaginar que uno está escribiendo el manual de usuario del sistema. A nadie se le ocurriría escribir un manual con un solo capítulo que describa toda su funcionalidad. De la misma forma no se debe escribir una especificación con un solo caso de uso.

Un caso de uso debe detallarse mediante una descripción textual. Los casos de uso se documentan con texto informal. En general se usa una lista numerado de los pasos que sigue el actor para interactuar con el sistema

Puedes usar una plantilla similar a la siguiente.

> ***Caso de uso***: <Nombre del caso de uso>
> ***Actor***: <lista de actores>
> <Lista de pasos a seguir>

Ejemplo: Mostraremos una simplificación de la descripción del caso de uso “Ingresar Pedido”.

Caso de uso: Ingresar Pedido
Actor: Empleado de Ventas
1) El cliente se comunica con la oficina de ventas. e informa su número de cliente
2) El encargado de ventas ingresa el número de cliente en el sistema
3) El sistema obtiene la información básica sobre el cliente.
4) El cliente informa el producto que quiere comprar, indicando la cantidad
5) El sistema obtiene la información sobre el producto solicitado y confirmar su disponibilidad
6) Se repite el paso 4) hasta que el cliente no informa más productos.
7) …...

Los casos de uso presentan varias limitaciones

- ¿Cómo hago un caso de uso para especificar el comportamiento interno del sistema? Debemos usar una nueva notación algo equivalente a los diagramas de flujo de datos del análisis estructurado. UML propone usa una notación llamada “Diagrama de Actividad”
- Otra limitación es que no hay una sintaxis clara dentro de la descripción del caso para indicar las decisiones y las iteraciones
    - Es frecuente recurrir a frases como: “se repite el paso X hasta que ocurre C” o “ Si ocurre C se pasa al paso X”.
    - En estos casos lo importante no es la forma en que se expresen las condiciones e iteraciones sino hacerlo de forma consistente.

ACTIVIDAD 1. Pon como ejemplo un sistema informático que pudieran encargarte desarrollar y enumera los posibles casos de uso que estarían presentes. No es necesario que los implementes con ningún diagrama ni que realices una descripción textual. Tan solo te pido un listado, lo más extenso posible. Puedes usar alguno de los supuestos que durante el curso hemos propuesto en otros temas.

ASOCIACIONES

Una asociación entre un actor y un caso de uso se da cuando el actor interactúa con el sistema para llevar a cabo el caso de uso

Las asociaciones no son obligatorias. Si en un diagrama de casos de uso aparece una asociación entre un actor y un caso, indica que "***puede***" que ese actor interactúe con el sistema en ese caso de uso

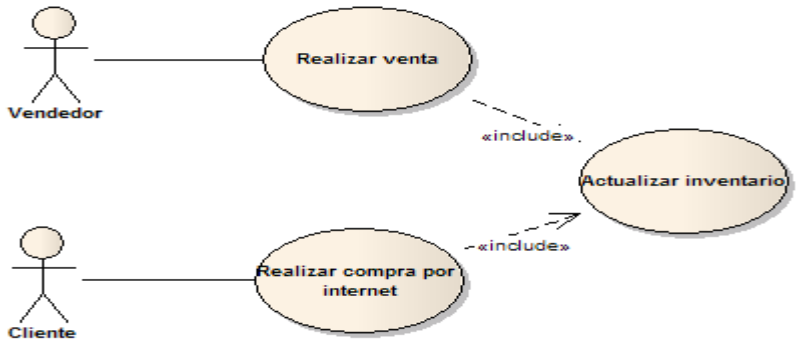
Tipos de asociaciones

Las podemos clasificar en:

1. Include
2. Extend
3. Generalización.

1. ***INCLUDE***

Se emplea para especificar un comportamiento común a dos o más casos de uso



- La punta de la flecha se encuentra en el caso de uso con el comportamiento común

Es frecuente que la misma funcionalidad del sistema sea accedida a partir de varios casos de uso. Para evitar repetir el texto, en la descripción textual, de esta funcionalidad en todos los casos de uso que la acceden puedo sacar esa funcionalidad a un nuevo caso de uso, que es usado por los casos de uso de los cuales fue sacada.

- Este concepto es similar al de subrutina o subprograma.

Las características de estas relaciones son

1. Aparecen como funcionalidad común, después de haber especificado varios casos de uso
2. Los casos <<include>> son casos de uso en sí mismos.
3. El caso es usado siempre que el caso que lo usa es ejecutado. Esto marca la diferencia con las extensiones (<<extend>>)

Las ventajas de estas asociaciones son:

- Las descripciones de los casos de uso son más cortas y se entienden mejor
- La identificación de funcionalidad común puede ayudar a descubrir el posible uso de componentes ya existentes en la implementación.

Las desventajas son:

- La inclusión de estas relaciones hace que los diagramas sean más difíciles de leer.

ACTIVIDAD 2. Pon como ejemplo un caso de uso que conste de una o varias asociaciones <<include>>. Debes expresarlas usando la notación adecuada.

- **El caso de uso no debe ser especificado textualmente pero sí gráficamente.**

---

2. EXTEND

Se especifican variantes del mismo caso de uso, es decir, el comportamiento del caso de uso es diferente según las circunstancias.

Estas variantes pueden mostrarse como diferentes descripciones o escenarios asociados al mismo caso de uso

(La fecha va hacia el caso de uso original)

Las extensiones tienen las siguientes características

- Representan una parte de la funcionalidad del caso que no siempre ocurre
- Son un caso de uso en sí mismo
- No necesariamente provienen de un error o excepción.

Llegados a este punto conviene hacer una digresión para introducir el concepto de alternativa.

Alternativas

Durante la ejecución de un caso de uso suelen aparecer errores o excepciones. Estas desviaciones del curso normal del caso de uso se llaman alternativas. Las alternativas tienen las siguientes características.

1. Representan un error o excepción en el curso normal del caso de uso.
2. No tienen sentido por sí mismas, fuera del contexto del caso de uso en que ocurren.

En la bibliografía las alternativas se documentan al final del caso de uso pero una alternativa es documentar los casos en tablas, mostrando el curso principal en la primera columna, y las alternativas en una segunda columna.

Ejemplo

Caso de Uso: Ingresando Pedido	
Actor: Empleado de ventas	
Curso Normal	***Alternativas***
1) El cliente se comunica con la oficina de ventas y proporciona su número de cliente.	
2) El encargado de ventas ingresa en número de cliente en el sistema	
3) El sistema obtiene la información básica sobre el cliente.	3.1- Si no está registrado, se le informa que debe registrarse.
4) El cliente informa el producto que quiere comprar, indicando la cantidad	
5) El sistema obtiene la información sobre el producto solicitado, y confirma su disponibilidad.	5.1.- Si no hay disponibilidad del producto, el sistema informa cuál será la fecha de reposición.
6) Se repite el paso 4) hasta que el cliente no solicite más productos	
……..	

Con este procedimiento es más fácil ver en qué parte del caso de uso puede ocurrir la excepción y se mantiene la posibilidad de ver el transcurso normal del caso de uso.

Llegados a este punto de la discusión hay una cuestión evidente. ¿ Cuál es la diferencia entre una alternativa y una extensión? .

- Una extensión es un caso de uso en sí mismo, mientras que una alternativa no.
- Una alternativa es un error o excepción, mientras que una extensión puede serlo o no

En la práctica pueden aparecer dudas sobre la conveniencia de considerar algo optativo en un caso de uso como una alternativa o una extensión sobre todo si no resulta claro si puede ser visto como un caso de uso en sí mismo o no.

Regla (aproximada): Podemos pensar que si algo opcional debe ser expresado con más de un paso, seguramente es una extensión no una alternativa

---

ACTIVIDAD 3. Pon como ejemplo un caso de uso que conste de una o varias asociaciones <<extend>>. Debes expresarlas usando la notación adecuada.

- **El caso de uso no debe ser especificado textualmente pero sí gráficamente**

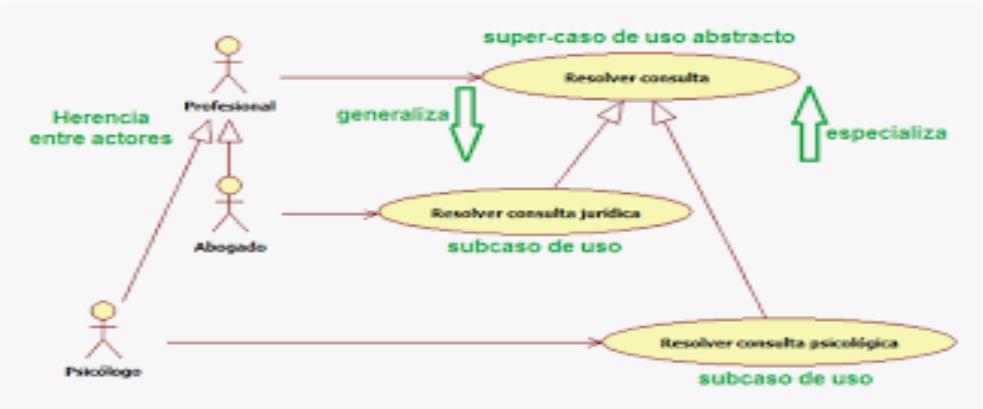
ACTIVIDAD 4 .Realiza la descripción textual de un caso de uso distinto a los anteriores. Intenta que dicha descripción sea lo más completa posible incluyendo alternativas

---

3. GENERALIZACIÓN.

En un diagrama de caso de uso también pueden mostrarse generalizaciones (relaciones de herencia) para mostrar que diferentes elementos están relacionados como tipos de otros. Son aplicables a actores o casos de uso.

- En el caso de los casos de uso su significado es similar a las relaciones <<extend>>



ESCENARIOS

Los escenarios son instancias del caso de uso. El caso de uso es lo general y el escenario es lo particular.

- ***Un caso de uso es un conjunto de escenarios con un mismo objetivo para el usuario. Podemos ver un escenario como uno de los muchos flujos que puede tener un caso de uso***

Todos los escenarios de un caso de un caso de uso deben tener en común que son intentos de hacer esencialmente lo mismo.

- Posteriormente los escenarios deben documentarse mediante diagramas de secuencia.

Ejemplo: En el caso de uso llevarse un libro en préstamo podemos encontrarnos, entre otros, con los siguientes escenarios.

- **Escenario 1**: José García se lleva prestado el tercer ejemplar de “Guerra y Paz” que hay en la biblioteca. No tiene ningún otro libro en préstamo
- **Escenario 2**: Mónica Díaz intenta llevarse prestado el primer ejemplar de “Ana Karenina”, pero no puede porque ya tiene tres libros en préstamo, que es el máximo

Si hemos documentado “paso a paso” el caso de uso debería sernos fácil identificar los distintos escenarios. La documentación de un escenario puede llevarse a cabo mediante una “plantilla” similar a la siguiente.

ESCENARIO: “Nombre del caso de uso al que se refiere”
Numeración:<Número del paso del caso de uso.[número alternativa]
Precondición: <Condiciones que dan lugar a que lleve a cabo este escenario
Postcondición:<Situación en que queda el sistema tras llevarse a cabo el escenario>
Quién lo comienza:<Actor que inicia el escenario>
Quien lo finaliza:<Actor que finaliza esta instanciación del caso de uso>
Excepciones:<Excepciones que puedan darse a la ejecución de este escemario>
Descripción:<Descripción detallada de las acciones que se llevan a cabo>

Observación: No todos los puntos deben completar la descripción de cada escenario. Tan solo se complementan si se considera necesario para su comprensión. No olvides que el objetivo del modelado debe ser realizar una descripción clara y comprensible a la par que rigurosa del sistema que pensamos construir, pero no debes sacrificar “comprensión” en aras del rigor

EJEMPLO.

ESCENARIO:"Llevar prestado un libro"
Numeración 1.2
Precondición: -
Postcondición: -
Quien lo comienza: SocioBiblioteca
Quien lo finaliza: Bibliotecario
Excepciones: Si el usuario quiere realizar el préstamo o no es un socio de la biblioteca sino parte del personal de la misma, el número máximo de libros en préstamo aumentan hasta 12.
Descripción: El socio de la biblioteca intenta llevarse prestado el primer ejemplar de un libro. El sistema comprueba que: ● Esa persona es socia de la biblioteca ● Los libros que ya tiene en préstamo no sobrepasan el número ,máximo de libros que se puede tener - 1 libro Debe tenerse en cuenta que el máximo número de libros en préstamo es de 6. Si las anteriores comprobaciones son correctas, el sistema comprueba si el ejemplar que se desea llevar está reservado por otra persona. Si el ejemplar está reservado, entonces no se permite el préstamo. Si no lo está, el sistema registra el préstamo y avisa al bibliotecario de que se debe anotar en el libro la fecha máxima de devolución (que aparecer por pantalla)

ACTIVIDAD 5.Realiza la descripción textual de un escenario correspondiente al caso de uso de la actividad anterior . Intenta que dicha descripción sea lo más completa posible.

CASOS DE TRAZO GRUESO vs CASOS DE TRAZO FINO.

En el modelo de ciclo de vida de desarrollo de sistemas llamado “modelo incremental” se van entregando versiones parciales del sistema, que implementan una parte de su funcionalidad. Se identifican todos los requerimientos que se puedan, sus propiedades y seleccionar cuáles se van a ir implementando en cada versión.

Para aplicar los casos de uso a desarrollos incrementales, empezamos por identificar todos los casos de uso del sistema, solo al nivel de su nombre. Una vez que los identificamos los expresamos en “trazo grueso” , esto es

- Ignoramos detalles sobre la forma de interacción entre el actor y el sistema
- Solo incluimos las alternativas más relevantes
- No entramos en detalle sobre las acciones que realiza el sistema cuando el usuario interactúa con el. Por ejemplo si el sistema tiene una política de descuentos para sus clientes, no es necesario especificar cómo es esa política: nos es suficiente con saber que existe y que debe ser tenida en cuenta

De esta forma obtenemos una descripción “gruesa” de todos los casos de uso

Los casos de uso de “trazo fino” son aquellos que se especifican una vez que se ha tomado la decisión de implementarlos. En este momento se completan todos los detalles.

- Incluimos todas las alternativas. En particular especificamos todos los errores y excepciones.

HEURÍSTICA PARA LA ESPECIFICACIÓN DE CASOS DE USO

Puedes seguir los siguientes pasos para el descubrimiento y documentación de los casos de uso. Estas notas tan solo pretenden ayudarte en tu trabajo. En ningún caso pretende “sentar cátedra” , puedes usar tu propia metodología.

1. Identificar a todos los actores que intervienen
2. Identificar todas las tareas que realiza cada actor
3. Agrupar tareas repetidas
4. Generar el diagrama o diagramas UML que represente esquemáticamente los casos de uso
5. Por cada caso de uso generar una especificación textual que lo documente.

Para la documentación textual puedes usar una plantilla similar a esta.

Caso de Uso: "Nombre del caso de uso"	
Actor(es): <listado de los actores que interactúan con el caso de uso>	
Curso Normal	***Alternativas*** (*)
1) <Descripción operación 1>	[1.1-Alternativa o errores operación 1] [1.2-Alternativa o errores operación 1] …..
2) <Descripción operación 2>	[2.1-Alternativa o errores operación 2] [2.1-Alternativa o errores operación 2]
…….	……..

(*) Las alternativas y errores pueden estar o no presentes (por ello hemos usado la notación, típica del pseudocódigo [] que representa las alternativas)

ACTIVIDAD 6. Realiza el siguiente cuestionario. Tu profesor te puede pedir que justifiques tu respuesta, ten en cuenta este punto cuando selecciones tu respuesta.

1. **¿Qué describen los casos de uso?**
2. **Los casos de uso corresponden a un conjunto de interacciones entre un usuario y el sistema**
    a. **Si**
    b. **No**
3. **Un caso de uso tiene en cuenta los objetivos no funcionales de un usuario**
    a. **Si**
    b. **No**
4. **En un caso de uso, un actor representa un usuario que representa una función precisa en el uso del sistema.**
    a. **Si**
    b. **No**
5. **Para los actores primarios, el objetivo del caso de uso es esencial.**
    a. **Si**
    b. **No**
6. **Para los actores secundarios, el objetivo del caso de uso también es esencial**
    a. **Si**
    b. **No**
7. **Un actor es una persona interna al sistema**
    a. **Si**
    b. **No**
8. **Un actor es obligatoriamente una persona física**

   a. Si
   b. No
9. La relación de comunicación relaciona un actor con el sistema
   a. Si
   b. No
10. ¿Cuál es la meta de las relaciones de inclusión y extensión?. ¿En qué se diferencian?
11. ¿Todos los casos de uso tienen una relación de comunicación directa con un actor?
   a. Si
   b. No
12. La relación de generalización/especialización es una relación que relaciona dos casos de uso
   a. Si
   b. No
13. Durante el desarrollo de un proyecto ¿ En qué momentos se utilizan los casos de uso?

EJERCICIOS Y SUPUESTOS

1.- El sistema consiste en una lámpara de noche. Las dos interacciones relacionadas con la funcionalidad del sistema son el encendido y el apagado. El actor primario es el usuario de la lámpara. Para este el encendido y apagado de la lámpara es su objetivo esencial

- Representa en UML los dos casos de uso; encendido o apagado de la lámpara
- Incluye la red eléctrica en el caso de uso del encendido de la lámpara

2..- Un hipódromo ofrece a sus clientes la posibilidad de asistir a las carreras y de realizar apuestas.
¿ Cuáles son los actores que interactúan con estos servicios?
Construye el diagrama de casos de uso.

3. - Describe el caso de uso de la autenticación de un usuario en un sistema informático. Esta autenticación se efectuará de manera simple con la introducción de un nombre y contraseña.

- Representa en UML el caso de uso de la autenticación con el actor que interactúa con este último caso sin especificar detalles.
- Introduzca en el diagrama del caso de uso el nombre y la contraseña, así como la verificación de estos datos
- Añada la introducción de un código complementario después de la contraseña. Este código complementario es opcional y solo está destinado a los usuarios que necesiten una seguridad reforzada

4.- El objetivo es describir la funcionalidad de un sitio de Internet para la consulta de horarios de trenes con la opción de poder comprar un billete correspondiente al horario seleccionado.

- Representa en UML el caso de uso de consulta con la descripción de la petición (elección de las estaciones de salida y llegada, fecha y hora de salida, posibilidad de tener una estación intermedia con una posible espera y la visualización de los horarios calculados por el sistema. Para la elección de las estaciones es posible consultar una base de datos que contenga todas las estaciones de la red de ferrocarriles)
- Introduzca la posibilidad de comprar un billete correspondiente al horario previamente seleccionado entre los que se muestran. No se olvide de que el sistema solicitará la tarifa elegida para el billete. No describa los detalles de la elección de la tarifa, del pago ni tampoco del envío del billete.
- Representa el caso de uso visualización y selección de un horario bajo la forma de textual

5.- Una sociedad micológica desea desarrollar una aplicación informática que permita a sus socios realizar el ***reconocimiento*** macroscópico ***de*** las típicas ***setas*** con sombrero in situ donde se recojan, lo más fácil y seguro posible de tal forma que se minimicen las posibilidades de cometer errores en dicho reconocimiento evitando así las posibles intoxicaciones que en algunos casos puede ser incluso mortales. Para ello propone realizar un protocolo de reconocimiento desde dos puntos de vista distinto. Un primer protocolo, llamado reconocimiento automático y un segundo protocolo llamado reconocimiento manual

- Reconocimiento automático: el usuario va a introducir varias fotos de la seta de la que quiere realizar el reconocimiento . Estas fotos las introducirá el usuario en el sistema. Las fotos que se introduzcan serán de distintos elementos de la seta como: fotos del sombrero de la seta, fotos del himenio de la seta , fotos de pie de la seta , foto del anillo del pie de la seta y fotos de la volva de la seta. Es posible que de algunos de estos elementos no sea necesario introducir las fotos correspondientes al no existir dicho elemento. También es posible introducir varias fotos de cada elemento y no solo una.

  El sistema una vez recogido los datos de las fotos de los distintos elementos de la seta comienza a comparar dichas fotos con su base de datos mediante un algoritmo de reconocimiento de patrones, comenzado por el sombrero, en segundo lugar llega al himenio, después al pie, después el anillo del pie si lo hubiese y por último la volva- El objetivo final es determinar el género y so es posible la especie de dicha seta.

  Una vez procesada la información si el sistema no puede determinar el género y la especie propondrá al usuario que introduzca más fotos o finalice el reconocimiento. Este último proceso será repetitivo hasta que el sistema consiga reconocer perfectamente el género y la especie de la seta en cuestión. En ese momento el sistema propondrá diferentes información al usuario aunque no siempre esta información es completa. La información es la siguiente: información del género al que pertenece la seta, información de la especie a la que pertenece la seta, informacion de fotos de la seta en distintos estados si la hubiera, información de toxicidad de la seta si la hubiera , información de comestibilidad de la seta si la hubiera, información sobre posibles confusiones con otras setas si la hubiera y por ultimo recomendación sobre la seta si las hubiera.
- Reconocimiento manuka: en este tipo de reconocimiento el sistema comenzará realizando preguntas al usuario y este deberá ir contestando en el orden en el que se realicen. Al comienzo el sistema estará esperando que el usuario pulse la tecla de comienzo de reconocimiento de la seta.Cuando esto suceda el sistema comienza preguntado por el tipo de himenio que tiene la serta. lo que el usuario contesta eligiendo de una lista una de las opciones posibles. Si la respuesta es “Por pliegues” el sistema comenzará a procesar esta información para descartar todas las setas que no tienen este himenio. En estos momentos el sistema pregunta al usuario si los pliegues están bien marcados o no, tanto con una como con otra respuesta por parte del usuario el sistema procede a reconocer las era y siempre en ambos casos es reconocido con éxito volviendo el sistema al estado inicial de espera. Si la respuesta anterior no es “Por pliegues” sino que es “Por agujas” el sistema volverá a procesar la información para descartar todas las setas que no tiene ese himenio. En estos

momentos el sistema preguntar si la carne de la seta es consistente es frágil, tanto con una como con otra respuesta por parte del usuario el sistema procede a reconocer la seta y siempre en ambos casos es reconocida con éxito volviendo el sistema al estado iniciar de espera- Cualquier otro tipo de himenio el sistema volverá al estado inicial de espera.

Se desea realizar el diagrama UML de casos de uso general solo del reconocimiento automático

6.- Se desea desarrollar una aplicación de ***gestión fincas e inmuebles***. La aplicacion debera cubre todos los aspectos relacionados con dicho tema, teniendo en cuenta la siguiente dinámica de funcionamiento:

Una empresa gestiona un conjunto de inmuebles que administra en calidad de propietaria. Cada inmueble puede ser bien un local (local comercial, oficina, ..), un piso o bien un edificio que á su vez tiene pisos y locales. Como el número de inmuebles que la empresa gestiona no es un número fijo, la empresa propietaria exige que la aplicación permite tanto introducir nuevos inmuebles con sis datos correspondientes (dirección, número, código postal, …) así como datos de baja , modificarlos y consultarlos. Asimismo, que una empresa administr un edificio determinado no implica que gestiones todos sus pisos y locales, por lo que la aplicación también deberá permitir introducir nuevos pisos o locales con sus datos correspondientes (planta, letra, ..), darlos de baja ,modificarlos y hacer consultas sobre ellos

Cualquier persona que tenga una nómina , un aval bancario, un contrato de trabajo o vende avalado por otra persona puede alquilar el edificio completo o alguno de los pisos o locales que no estén ya alquilados y posteriormente desalquilar. Por ello deberan poderse dar de alta si no nuevos inquilimos con suis datos corspondientes (nombrem DNI m edad ,sexo fotografia, …) poder modifcarlos datos de baja etc (par la realizacion de cualqueira de stas oepaiones es necsaira la identificacion por aprte del inquilno)

Por otra parte, cada mes el secretario de la empresa pedirá la generación de un recibo para cada uno de los pisos y de los locales. el cual lleva asociado un número de recibo que es único para cada piso y para cada local y que no variará a lo largo del tiempo. indicando el piso o local a que pertenece, la fecha de emisión, la renta. el agua , la luz, la actualización de IPC anual, portería, IVA, etc. Y otros obligatorios (para todos los recibos). Además , para cada recibo se desea saber si está o no cobrado.

Con vistas a facilitar la emisión de recibos cada mes, la aplicación deberá permitir la generación de recibos idénticos a los del mes anterior a excepción de la fecha. Además, deberán existir utilidades para inicializar los conceptos que se desee de los recibos en una determinada cantidad y también debe ser posible modificar recibos emitidos en meses anteriores al actual. La aplicación también deberá presentar los recibos cuyo importe sea igual a cero.

De igual forma el secretario debe poder gestionar los movimientos bancarios que se producen asociado a cada edificio, piso o local. Un movimiento bancario siempre está asociado a un bando y una cuenta determinando de ese bando. En esa cuenta existirá un saldo acreedor o deudor, que aumentará o disminuirá con cada movimiento. Para cada movimiento que se desea saber también la fecha en que se ha realizado. Un movimiento bancario puede ser de dos tipos: un gasto o un ingreso.

Si el movimiento bancario es un gasto entonces estará asociado a un inmueble determinado, se indicará el tipo de gasto al que pertenece entre los que se tiene estipulados. Ejemplos de gastos son los costes de la reparación de un ascensor del inmueble que pertenece a los gastos de reparación, el sueldo de la señora de la limpieza, etc. Si el movimiento bancario es un ingreso entonces estará asociado a un piso de un inmueble determinado o a un local y también se indica el tipo de ingreso al que pertenece como en el caso de los gastos. Ejemplo de ingreso son precisamente los recibos que se cobran cada mes a los inquilinos.

Basándose en los gastos e ingresos que se deducen de los movimientos bancarios, la aplicación deberá ser capaz de ocuparse de la gestión económica generando los informes que facilitan la realización de la declaración del arenta.

Por último la aplicación deberá ser capaz de proporcionar el acceso, de forma estructurada , a toda la información almacenada en el sistema, generando para ello los listados necesario que quiera el secretario

Ejemplos de listados son_: listado de todos los inquilinos ordenado por fecha, el listado de inquilino que han pagado o no un determinado intervalo de tiempo el listado de todos los inmuebles , el listado de todos los pisos y locales de cad edificio., el listado de todos los recibos pendiente de cobro en un determinado intervalo de tiempo, etc.

7.- Se desea desarrollar una ***aplicación de gestión de las calificaciones*** de los alumnos para satisfacer las innumerables quejas de los profesores, por el uso del lápiz y papel. La aplicación deberá cubrir únicamente aquellos aspectos relacionados con dicho tema, y que se describen a continuación:

El profesor recibe las actas en blanco de las asignaturas de las que es responsable, en formato electrónico. El acta contiene los siguientes datos de la asignatura (titulación, campus. curso académico, denominación de la asignatura, convocatoria y grupo) y la lista de alumnos matriculados (núm exp, nif, nombre y apellidos). Alguna de las acciones que pueden hacer el profesor son:

- Completar un acta con las notas de los alumnos
- Añadir y borrar un alumno de un acta
- Integrar las actas de varios grupos de una misma asignatura en una sola acta

Otras de las opciones que se le exige a la aplicación, para satisfacer completamente las necesidad del profesor son las siguientes

- Permitir la consulta de la siguiente información de cualquier alumno seleccionado.
    - DNI, NO EXPEDIENTE. Lista de asignaturas en las que está matriculado el alumno (Código asignatura-nombre asignatura)
- Obtener una estadística de las calificaciones obtenidas por los alumnos en un determinado grupo de una asignatura. En esta estadística se tendrá para cada posible calificación
    - Número de personas con esa calificación. Porcentaje sobre los presentados. Porcentaje sobre el total del grupo.
- Consultar el porcentaje de personas sobre el total del grupo que se han presentado y el de los que no se han presentado

- Poder visualizar un gráfico indicación del número de personas que han obtenido una calificación entre 0-0.99, 1-1.99, 2-2.99, 3-3.99. 4-4.99, 5-5.99, 6-6.99, 7-7.99, 8-8.99, 9-10; indicándose la nota media obtenida por clase.
- Disponer de una calculadora que permita realizar las operaciones de suma, resta, multiplicación, división. Está calculadora se activará cuando se vayan a introducir las notas a algún alumno de forma que una vez realizada la operación aritmética,

- pulsando un botón se vuelque el resultado en la casilla donde se están introduciendo las calificaciones , redondeando a dos cifras decimales.
- Permite la importación y exportación de las lista de alumnos con sus calificaciones a un formato compatible con MS Excel.
- Imprimir las actas y las listas provisionales de calificaciones.

Finalmente como una ampliación extra , a la cual podrá acceder quien se identifique inicialmente como administrador de la aplicación se deben permitir

- Gestión ABMC (altas/Bajas/Modificaciones y Consultas) de los datos de un alumno y su matriculación en una asignatura y a un grupo
- Gestión de Asignaturas, teniendo en cuenta que una asignatura sólo se puede dar en un único curso(primero, segundo, tercero, ..) y que cada curso está formado por los datos sobre el número máximo de alumnos, número mínimo de créditos troncales y número mínimo de créditos optativos. Algunos de los datos que vamos a poder consultar de una asignatura son el nombre , número de créditos y cuatrimestre en el que se imparte.
- Gestión de Titulaciones, teniendo en cuenta que una titulación solo se da en un campus determinado y los datos que podemos consultar son el nombre, el número de créditos o carga lectiva global, si es de 1º o 2º ciclo,..
- Gestión de grupos, en los que podemos consultar el número máximo de alumnos admitidos, si es un grupo de mañana. de tarde o de noche, y cual es el código empleado para identificar el grupo.
- Consultar aquellos alumnos que no se pueden matricular y el motivo de ello
- Consultar el historial académico de un alumno

Se desea realizar el diagrama UML de casos de uso.

BIBLIOGRAFÍA

- *Grady Booch, James Rumbaugh e Ivar Jacobson. El lenguaje unificado de modelado. Addison Wesley 2º edición.*

Referencia amplia sobre los elementos que componen el UML, además está "salpicada" de buenos consejos, fruto de la experiencia de los autores. Se "echa de menos" más ejemplos y una relación de ejercicios o supuestos para aplicar los conceptos. No obstante creo que es un de los mejores recursos para aprender el UML

- Grady Booch, James Rumbaugh e Ivar Jacobson. UML gota a gota. Addison Wesley

Otra obra de "los tres amigos". A pesar de su reducida extensión (menos de 200 páginas y en un formato algo menor que DIN A4) "vale su peso en oro". Es el manual que más me ha ayudado a entender el UML y su aplicación en el modelado de sistemas. Explicaciones claras y ejemplos útiles y comprensibles. No obstante he de advertirte que su notación está algo obsoleta y difiere algo de la empleada en estas notas que te he preparado (lo que hemos denominado <<Include>> los autores lo denominan <<Uses>>). Al igual que la obra anterior creo, es tan solo una opinión personal, que deberían haber incluido una relación de supuestos y /o ejercicios. A pesar de todo esto (carencia de ejercicios y supuestos para "entrenarse"y notación algo obsoleta) es el manual de UML que me llevaría a una isla desierta (en el supuesto que de verme obligado a vivir en soledad el resto de mis días se me ocurriera llevarme como compañía un texto sobre UML y la orientación al objeto).

- Laurent Debrauwer y Fien Van der Heyde. UML 2 Ediciones ENI (Colección Informática Tècnica)

Está en las antípodas de las dos obras anteriores: descripción superficial de los tópicos del UML y la orientación al objeto, explicaciones poco claras en muchos casos y en otros toman decisiones de diseño cuanto menos sorprendente y que no justifican (o lo hacen de manera insuficiente en muchos casos). La razón de su inclusión en estas notas bibliográficas es doble.

- La inclusión de ejercicios prácticos
- La notación empleada que coincide con la que suele emplearse al día de hoy y por ende con la que yo he usado al redactar estos apuntes.

- Laurent Debrauwer y Naouel Karam. UML 2 Practique la modelización. Ediciones ENI (Colección Prácticas Técnicas)

Es el compañero necesario del libro anterior y el complemento de los dos primeros. Es un libro de supuestos y problemas, la mayoría muy sencillos y de poca utilidad en el mundo real pero que sirven para afianzar los conceptos del UML y el modelado orientado al objeto. Encontramos en él los mismos defectos y carencias que el obra anterior y al igual que

dicha otra se incluye por el número (debo reconocer que bastante extenso) de ejercicios y cuestiones teóricas. Por si te sirve de ayuda cuando comencé a estudiar el UML lo hice con las dos primeras obras (referencias de gran valor y que te aconsejo obtengas si tienes interés en aprender el profundidad sobre estos temas) y para *entrenarme* y *prácticar* me ayudé, en los primeros momentos, por estos dos últimos libros. Te aconsejo que valores la posibilidad de seguir la misma estrategia. Además de un precio reducido puedes acceder a ellos, junto con la mayoría de la base bibliográfica de la editorial ENI, de manera online pagando una suscripción.

- Roger S. Pressman. Ingeniería del Software Un enfoque práctico. Mc Grawhill

Es la referencia que más he dudado en incluir. No es un libro de Análisis Orientado al Objeto sino de Ingeniería del Software. Trata por tanto de los distintos aspectos y metodología de dicha disciplina, tanto del análisis estructurado como del orientado al objeto. Además de muchos otros tópicos: recolección de requisitos, generación de casos de prueba, etc. Es un libro de texto (a nivel universitario) con todo lo que dichos libros deben tener: Ejercicios y supuestos de distintos niveles de dificultad, referencias bibliográficas abundantes, soporte web y muchos parabienes más. Si has estado atento a mis clases habrás observado que lo he mencionado en repetidas ocasiones. La edición que tengo en mi biblioteca es la 5º pero en estos momentos está publicada la 7º. No la he podido estudiar pero es muy probable que siga manteniendo la calidad de las anteriores ediciones. Es el libro al que recurro cada vez que tengo alguna duda o deseo información sobre algún tema relacionado con esta disciplina. Si tienes interés te aconsejo encarecidamente que lo obtengas. Es (y esto no es una opinión personal dado que es recomendado en todas las reseñas bibliográficas) una de las mejores y más completas obras sobre la Ingeniería del Software escritas en nuestra lengua.

- José Ignacio Peláez Sánchez, José Ignacio Gámez Jimenez y Jesús María Doña Fernadez. DUM:Desarrollo Unificado con Métrica. Universidad de Málaga.

Es mi última adquisición y debo confesar que aún no lo he leído tan solo lo he “ojeado”. No obstante, dado que me ha causado una buena impresión he querido recogerlo en esta reseña. También quiero compartir contigo las razones que me impulsaron a adquirlo y por que creo que nos puede ser útil

- Describe en profundidad los diagramas del UML
- Emplea el modelo de desarrollo Unificado, una de las metodologías más empleadas en la empresa (¿imaginas quiénes son los autores de uno de los mejores libros sobre esta metodología? ¡SI ¡ Nuestros “ tres amigos”, quienes además fueron los creadores del UML)
- Fusiona dicha metodología con MÉTRICA que es la metodología “oficial” de las administraciones públicas de nuestro país.
- Acompaña la descripción de los tópicos que el libro desarrolla con un ejemplo completo.

Espero encontrar tiempo para leerlo y/o estudiarlo. Si no satisface mis expectativas (y aunque ello sea un juicio de valor) entenderé que no debe figurar en esta breve reseña y no aparecerá en otras versiones de estos apuntes.

- Antonio de Amescua Seco. Análisis y Diseño Estructurado y Orientado a Objetos de Sistemas Informáticos. Mc Grawhill.

Este es junto con el segundo libro que he mencionado en estas notas bibliográficas el que más he estudiado. Es un libro fundamentalmente de supuestos. Al contrario que los del manual de Laurent Debrauwer y Fien Van der Heyde no son triviales. El nivel es de un primer curso de Análisis a nivel universitario. Las razones para que figure en nuestra biblioteca son varias entre ellas destacaría las siguientes.

- Una notación actual, similar a la que he empleado en mis apuntes.
- Supuestos de un nivel de complejidad suficiente para lograr una comprensión del modelado que permita pensar en modelar para el mundo real (siempre teniendo en cuenta el abismo que existe entre el mundo académico y el real donde las cosas nunca son tan sencillas, bonitas y elegantes como se muestran en las aulas).
- Un resumen de los conceptos y notación a emplear antes de comenzar con los supuestos.
- Todos los supuestos están desarrollados en su totalidad.
- Trata tanto en análisis estructurado como el orientado al objeto , incluso el desarrollo de modelos conceptuales de bases de datos.

El autor también ha publicado un manual teórico sobre Ingeniería del software (es de suponer que este manual al que nos referimos es el complemento del teórico). No obstante, a pesar de mis esfuerzos, no he logrado hacerme de él. Es presumible que esté descatalogado. Si logras encontrarlo (en formato digital o papel no dudes en hacermelo saber)

- Jordí Pradel Miquel y José Raya Matos. Análisis UML. UOC (Universidad Oberta de Catalunya)

No puedo decirte mucho sobre él. Fué uno de los manuales que empleé durante la preparación de las oposiciones pero me centré más en el estudio del *Pressman,* el *Amescua Seco* y el Booch (“el gota a gota”) . No obstante creo que es un buen texto:claro y con ejemplos. No son tan complicados como los del *Amescua Seco* y posiblemente más adecuados para un estudiante de ciclo formativo. Son los apuntes de una lección de una de las asignaturas relacionadas con la Ingeniería del Software que imparte dicha universidad. Está bien lograda para ser tan solo una de las lecciones de dicha asignatura, es bastante completa pero dado que se pretende que sea completada con el resto de unidades no es suficientemente para comprender el modelado y diseño orientado al objeto. Se centra tan solo en los diagramas. Si tienes oportunidad de hacerte del resto de las unidades sin lugar a dudas te lo aconsejo. No obstante si te servirá para profundizar en la notación y funcionalidad de los distintos diagramas del UML.

"No están todos los que son, pero sin son todos los que están"

Soy consciente de que hay otras muchas otras obras sobre el análisis y diseño orientado al objeto. Muchas de ellas más actuales y de una calidad similar (y posiblemente superior). No obstante, no me ha parecido sincero incluirlas en esta reseña bibliográfica por una única razón: *No las he estudiado o leido .* Hubiera sido fácil realizar una búsqueda en Google e incluir en esta reseña un listado con la bibliografía que sobre el tema se ha publicado o entra en el OCW de alguna universidad e incluir los libros que el profesor responsable de la asignatura aconseja a sus alumnos. Estoy seguro que habría sido de valor para vosotros, dado que los conocimientos y experiencias literarias de cualquiera de ellos es, sin lugar a dudas, muy superior a la mía. No obstante no he querido '*hablar por boca de ganso*' (perdonan lo manida de la expresión y su, posiblemente, vulgaridad) y aconsejar un manual del que no tengo un conocimiento directo. ¡ Espero no haber errado ¡

En los restantes temas que trataremos sobre el Análisis Orientado al Objeto sólo recogeré las reseñas que no figuren aquí y que me parezcan de interés. Quiero que seas consciente que estos manuales no son sobre Casos de Uso sino sobre Análisis y Diseño Orientado al Objeto (y en muchos casos van más allá). Por dicha razón pueden serte de utilidad durante el estudio de esta asignatura y en tu futuro profesional y/o académico. Tan solo los he incluido aquí por ser este el primer tema que sobre el Análisis Orientado al Objeto y el UML tratamos en esta asignatura. Espero te presten tanta ayuda como me han prestado (y siguen prestando) a mi.