UCA Universidad de Cádiz

Análisis Orientado a Objetos en UML

TEMA 2

Departamento de Ingeniería Informática
© Elena Orta

Tema 2. Análisis Orientado a Objetos en UML

Tema 2. Análisis Orientado a Objetos en UML

Contenido de este fichero

Objetivos específicos

El alumno debe ser capaz de:

- Describir el objetivo principal del análisis de sistemas.
- Enumerar las actividades del análisis de sistemas y describir brevemente cada una de ellas.
- Enumerar y describir brevemente los principios fundamentales del análisis de sistemas.
- Describir brevemente el contenido del Documento de Especificación de Requisitos según el estándar IEEE Std. 830.
- Definir qué es UML (Unified Modeling Language).
- Enumerar los principales diagramas de UML.
- Definir qué es un modelo de análisis y enumerar los principales modelos de análisis.
- Describir el objetivo del modelo de casos de uso.
- Definir qué es un caso de uso y un diagrama de casos de uso.
- Definir qué es un actor de un Caso de Uso y los tipos de actores.

Objetivos específicos

El alumno debe ser capaz de:

- Definir que es un escenario de un Caso Uso y los tipos de escenarios.
- Definir los distintos tipos de relaciones entre casos de uso e identificar cuando usar cada una de ellas.
- Describir la relación que existe entre Caso de Uso y colaboración de objetos.
- Realizar modelos de casos de usos de sistemas propuestos.

Tema 2. Análisis Orientado a Objetos en UML

Actividades del análisis

Objetivo del análisis de requisitos

- **Realizar un Documento de Especificación de Requisitos del Software (especificar qué tiene que hacer el sistema y no cómo desarrollarlo).**
- **El Documento de ERS es el resultado del trabajo conjunto de los analistas y de los clientes o usuarios.**

Actividades del análisis de requisitos

- **Determinación de requisitos.**
- **Análisis de requisitos.**
- **Especificación de requisitos.**
- **Validación de requisitos.**

Puede haber iteraciones y solapamientos entre las actividades, no tienen que realizarse necesariamente en secuencia.

Actividades del análisis

Actividad 1: Determinación de los requisitos

- Definición de los requisitos del software.
- Definición de los requisitos de las interfaces del software con el sistema y con el exterior.
- Se utilizan técnicas de recogida de información: entrevistas, cuestionarios, formularios, etc.

Actividad 2: Análisis de los requisitos

- Proceso de razonamiento sobre requisitos obtenidos (Técnicas).
- Detectar/resolver inconsistencias, coordinar requisitos relacionados, etc.

Actividades del análisis

Actividad 3: Especificación de los requisitos

- **Documentar los requisitos del sistema.**
- **Utilizar lenguaje natural, especificación formal, modelos, gráficos, etc.**

Actividad 4: Validación de los requisitos

- **Confirmación con los clientes/usuarios que los requisitos son válidos, consistentes, etc.**
- **Utilizar listas de comprobación.**

Tema 2. Análisis Orientado a Objetos en UML

Principios fundamentales del análisis

Métodos de análisis

- **Existen diferentes formas de realizar el análisis de un sistema software: análisis estructurado, análisis orientado a objetos, etc.**
- **Los métodos de análisis tienen puntos de vista, notaciones y técnicas distintas pero tienen principios fundamentales comunes.**

Principios fundamentales de análisis

- **Modelado**
- **Partición**
- **Planteamiento esencial de los requisitos**

Principios fundamentales del análisis

Modelado

Se realizan modelos desde tres perspectivas diferentes:

- **Función: ¿Qué hace el sistema?**
- **Información: ¿Qué información utiliza el sistema?**
- **Comportamiento: ¿Cuándo sucede algo en el sistema?**

- **Es importante analizar las relaciones que existen entre los distintos modelos.**
- **La importancia de cada modelo depende de las características del sistema y del método de análisis utilizado.**

Funciones de los modelos

- **Entender el sistema**
- **Son la base de la revisión del análisis y del diseño**

Principios fundamentales del análisis

Partición

- Dividir el problema en partes y establecer interrelaciones entre ellas para obtener la función global.

Planteamiento esencial de los requisitos

- Analizar las funciones y la información del sistema sin tener en cuenta los detalles de implementación.

Tema 2. Análisis Orientado a Objetos en UML

Especificación requisitos del software

Documento de Especificación de los Requisitos del Software (DERS)

Documento que contiene la especificación de los requisitos del software y de las interfaces externas:

- **Funcionalidad:** ¿Qué debe hacer el software?
- **Prestaciones:** Rendimiento, tiempo de respuesta, etc.
- **Restricciones de diseño:** Lenguaje de programación, recursos disponibles, etc.
- **Atributos:** Portabilidad, Corrección, Mantenibilidad, etc.
- **Interfaces** con los usuarios, con el hardware, con otros sistemas, etc.

Especificación requisitos del software

DERS no incluye:

- **Aspectos de diseño: estructura modular, estructuras datos, interfaz intermodular, etc.**
- **Aspectos de gestión de proyectos: costes, plazos de entrega, etc.**

El Documento de ERS evoluciona con el tiempo para:

- **Añadir requisitos no contemplados al comienzo del proyecto.**
- **Corregir deficiencias o errores en los requisitos.**

Estándar DERS

IEEE Std. 830. Recommended Practice for Software Requirements Specifications
(Estándar del Documento de Especificación de Requisitos del Software)

Tema 2. Análisis Orientado a Objetos en UML

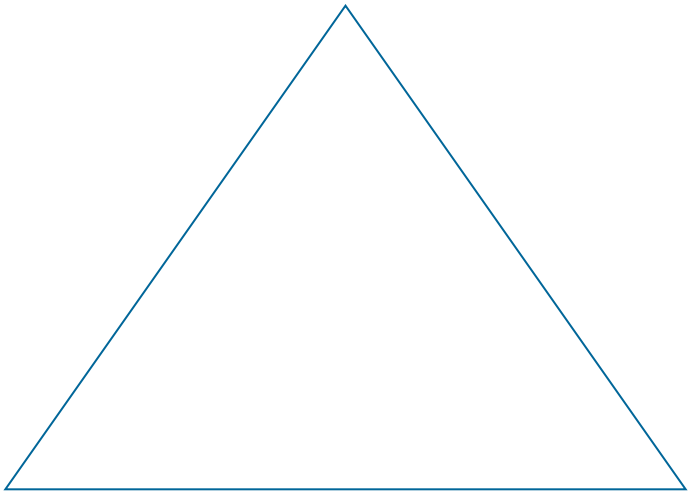
Análisis OO (UML)

UML es un lenguaje para visualizar, especificar, construir y documentar los modelos de un sistema desde una perspectiva OO.

- **Es una notación no es un proceso.**
- **Hay muchos procesos que utilizan UML como notación.**
- **Utilizable para sistemas que no sean software.**

Análisis OO (UML)

UML → Notación



Proceso (metodología)

- Unified Process
- Rational Unified Process
- Craig Larman

Herramienta

- Rational Rose, Objecteering, Visio
- Poseidon, Visual Paradigm, Umbrello, etc.

Principales diagramas UML

- Diagramas de Casos de Uso
- Diagramas de Clases
- Diagramas de Objetos
- Diagramas de Interacción
  - Diagramas de Secuencia
  - Diagramas de Colaboración

Los veremos en la asignatura

- Diagramas de Estados
- Diagramas de Actividades
- Diagramas de Componentes
- Diagramas de Despliegue

Modelos

- Un modelo es una simplificación de la realidad.
- Un modelo es el resultado de un proceso de abstracción y ayuda a comprender y razonar sobre una realidad.

Modelo software

- Un modelo software es una descripción de un aspecto del software expresada en un lenguaje bien definido.
- Se crean un conjunto de modelos software que permiten especificar aspectos del software como los requisitos, datos, comportamiento, etc.
- Los modelos:
  - ayudan a razonar sobre el sistema.
  - favorecen la comunicación.
  - permiten documentar las decisiones.
  - permiten una generación automática de código.

Modelos

- **Modelo de Casos de Uso.**
  - Diagramas de Casos de Uso.
- **Modelo Estructural.**
  - Diagrama de Clases.
- **Modelo de Comportamiento.**
  - Diagramas de Interacción.
  - Diagramas de Estados.
- **Modelo de flujos de Actividades.**
  - Diagramas de actividades.
- **Modelo Implementación.**
  - Diagrama de Componentes.
- **Modelo de Despliegue.**
  - Diagramas de Despliegue.

Modelos de análisis

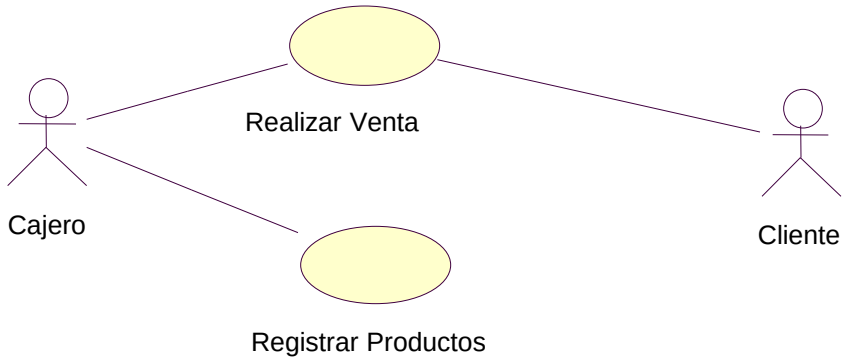
Modelo de casos de uso	Modelo conceptual datos	Modelo de comportamiento	Modelo de estados
Diagramas Casos de uso Casos de uso	Diagramas de clases (objetos del dominio del problema)	Diagramas de secuencia del sistema Contratos para las operaciones del sistema	Diagramas de estados de objetos y casos de uso

Modelos de análisis

Modelo de casos de uso	Modelo conceptual datos	Modelo de comportamiento	Modelo de estados
Diagramas Casos de uso Casos de uso	Diagramas de clases (objetos del dominio del problema)	Diagramas de secuencia del sistema Contratos para las operaciones del sistema	Diagramas de estados de objetos y casos de uso

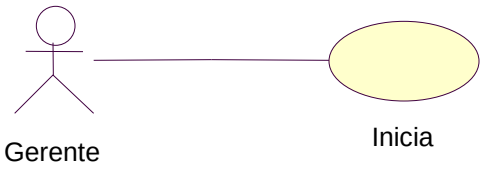
Los veremos en la asignatura

Modelo de casos de uso



Cu: Realizar Venta

Descripción: …
Actores: …
Precondiciones: …
Postcondiciones: …
Esc. Principal: …
Esc. Alternativos: …

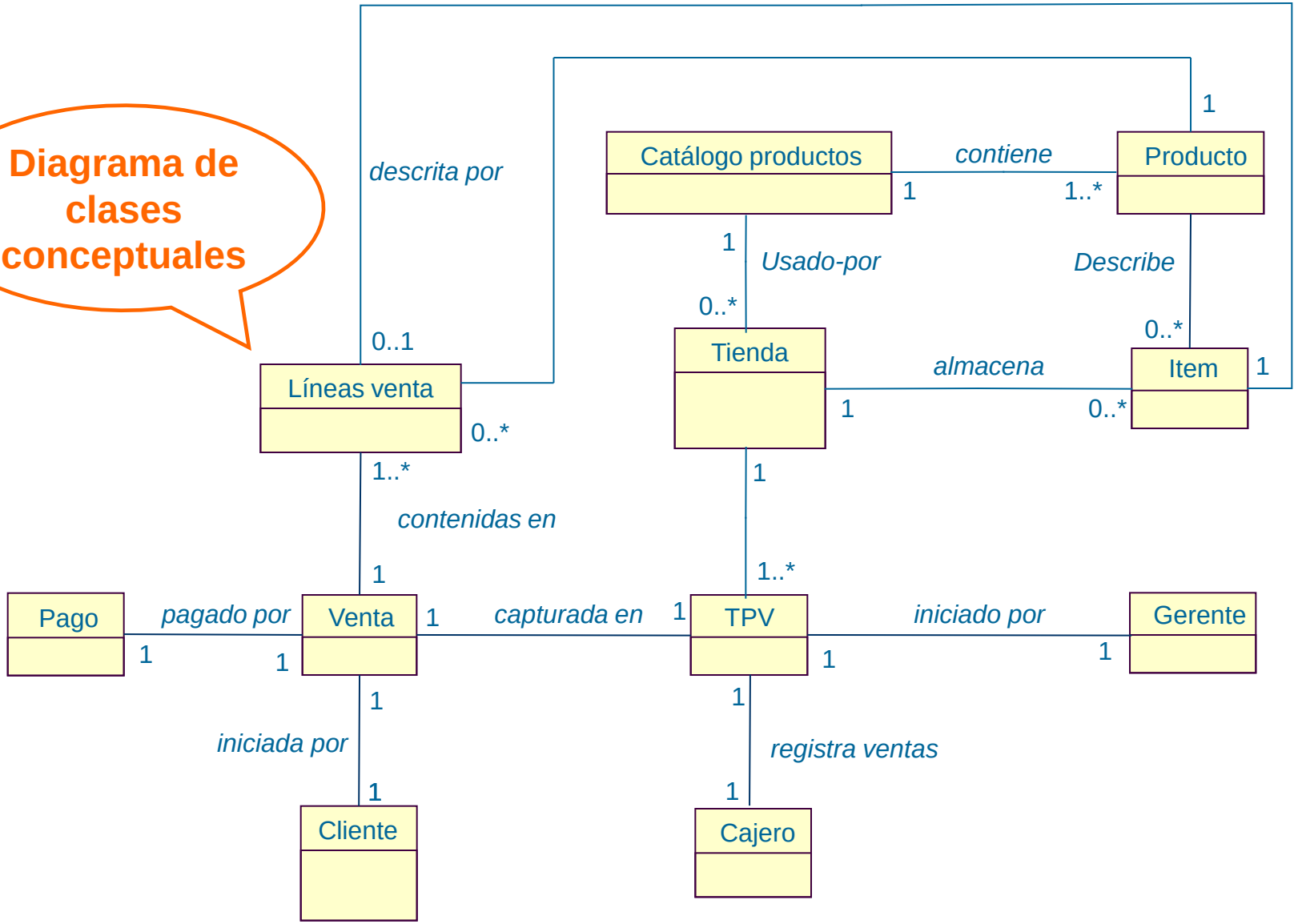


Descripción casos de uso

Diagrama de casos de uso

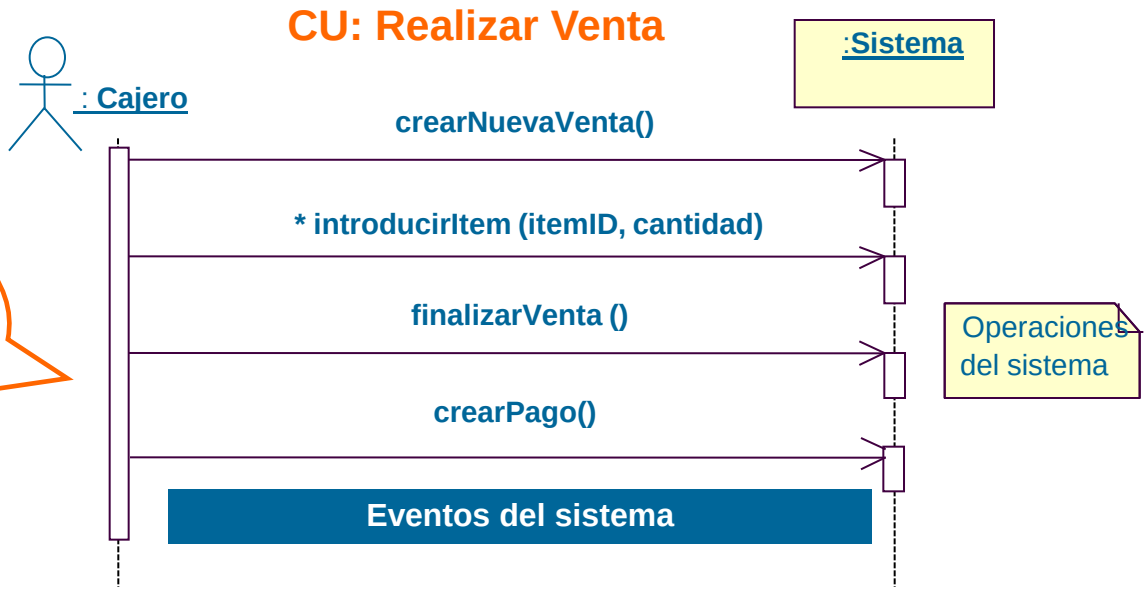
Modelo conceptual de datos (modelo de dominio)

Diagrama de clases conceptuales



Modelo de comportamiento

Diagrama secuencia de sistema

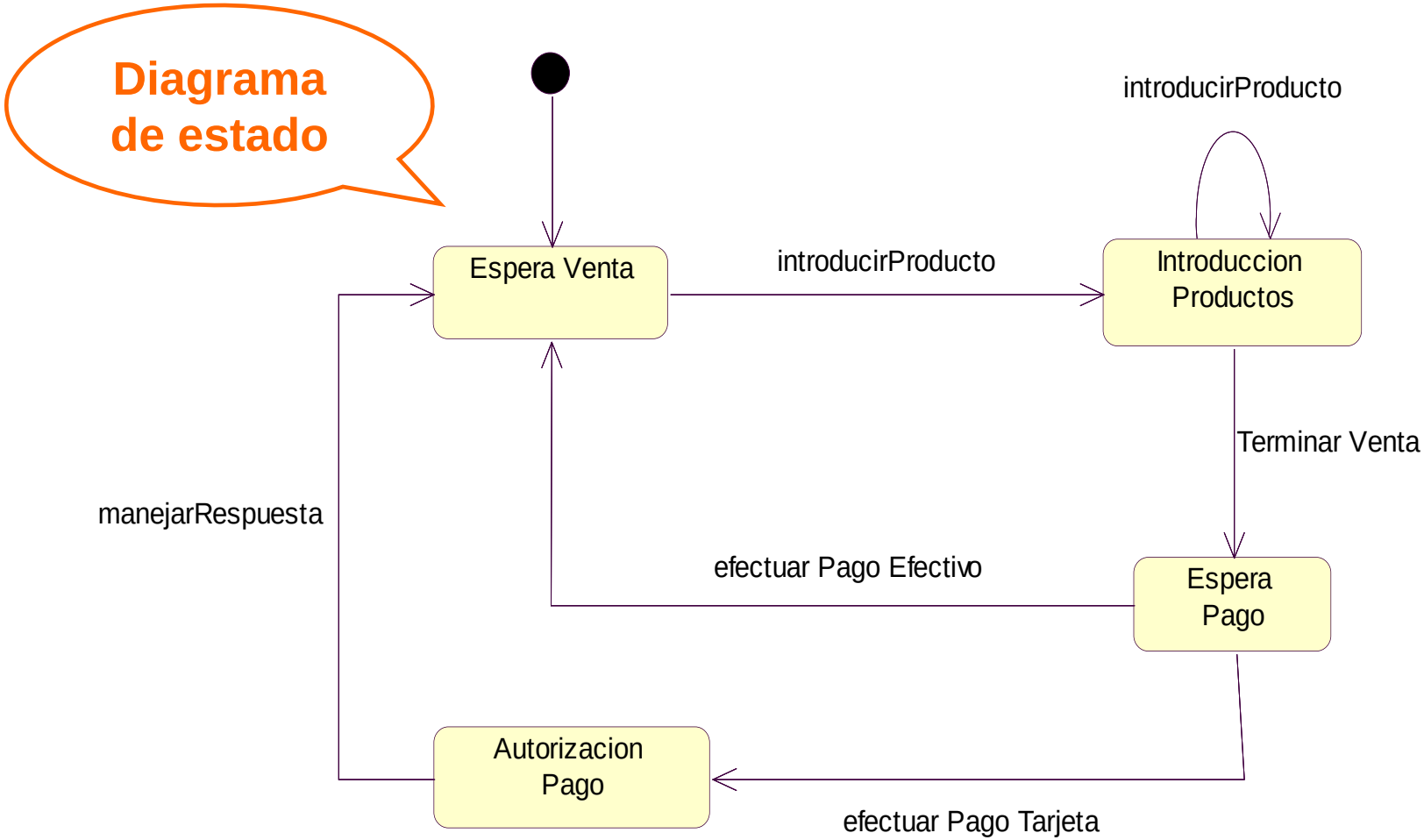


Contrato operación del sistema

Nombre: introducirItem (itemID: string, cantidad: integer)
Referencias Cruzadas: Registrar Venta
Precondiciones: Hay una venta en curso
Postcondiciones:

- Se creó una instancia *lv* de *LineaVenta*
- Se asoció *ldv* a la venta en curso *v*
- Se asignó *cantidad* a *lv.cantidad*
- *lv* se asoció a una *EspecificaciónProducto* según *itemID*

Modelo de estado



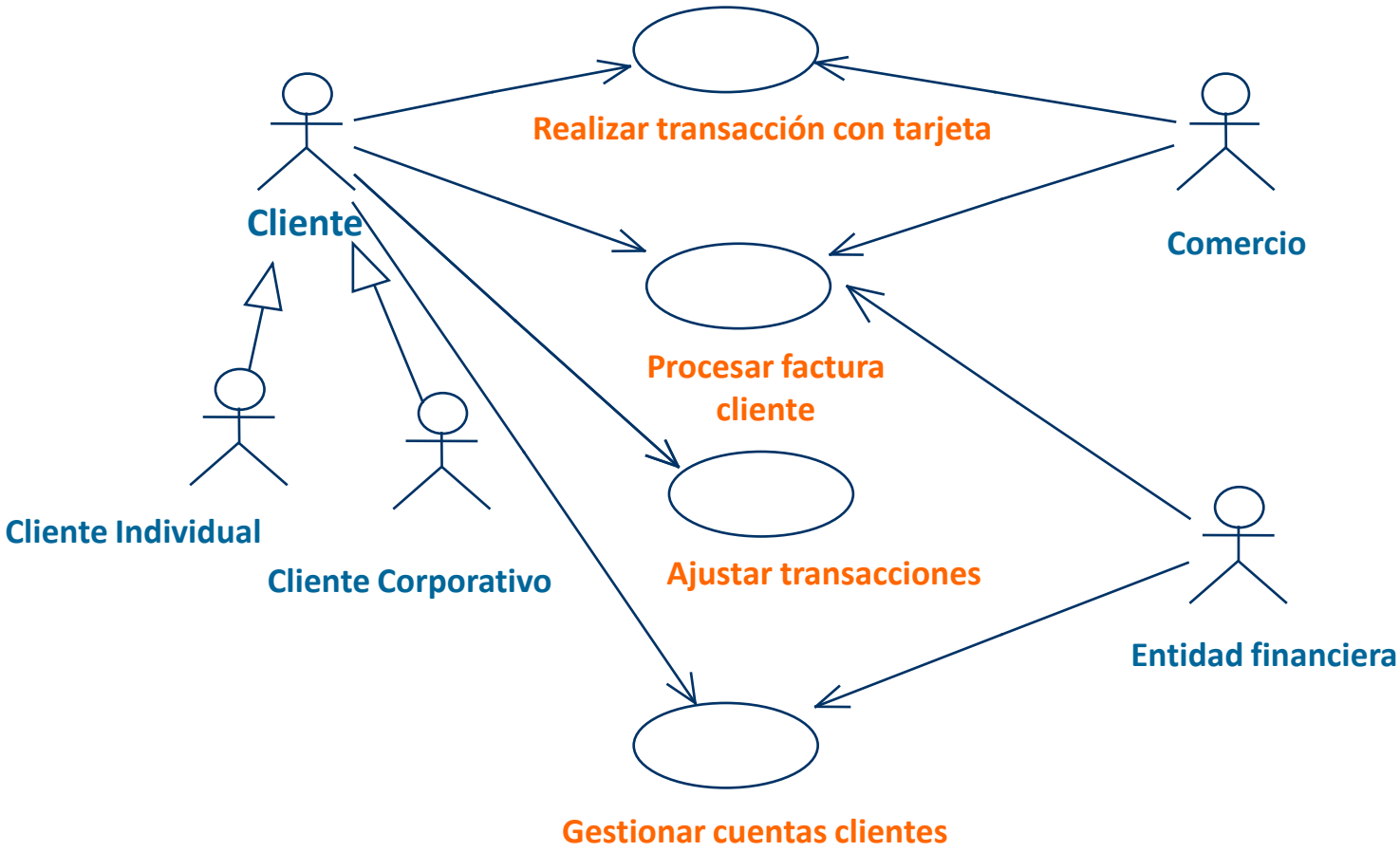
Tema 2. Análisis Orientado a Objetos en UML

Modelo de casos de uso

Modelo de casos de uso	Modelo conceptual datos	Modelo de comportamiento	Modelo de estados
Diagramas Casos de uso Casos de uso	Diagramas de clases (objetos del dominio del problema)	Diagramas de secuencia del sistema Contratos para las operaciones del sistema	Diagramas de estados de objetos y casos de uso

Diagrama de casos de uso

Representa Casos de Uso, Actores y Relaciones entre ellos



Casos de uso

- Especifican el comportamiento deseado del sistema.
- Ofrecen un medio sistemático e intuitivo para capturar los requisitos funcionales, centrándose en el valor añadido para el usuario.
- El conjunto completo de casos de uso especifica todas las posibles formas de usar el sistema.
- Dirigen todo el proceso de desarrollo puesto que la mayoría de actividades (planificación, análisis, diseño, validación, pruebas,..) se realizan a partir de los casos de uso.

Los Casos de Uso NO son Orientados a Objetos: Herramienta para el análisis de requisitos que se puede utilizar en proyectos no orientados a objetos

Casos de uso (definición)

- ***"Describe un conjunto de interacciones entre actores externos y el sistema en consideración orientadas a satisfacer un objetivo de un actor".***

- ***"Es una colección de posibles secuencias de interacciones entre el sistema en discusión y sus actores externos, relacionada con un objetivo particular".***

- ***"Un caso de uso especifica un conjunto de secuencias de acciones, incluyendo variantes, que el sistema puede ejecutar y que produce un resultado observable de valor para un particular actor".***

Casos de uso

- Son iniciados por un **actor** con un objetivo en mente y es completado con éxito cuando el sistema lo satisface.
- Puede haber **secuencias alternativas de interacciones** que llevan al éxito y/o al fracaso en la consecución del objetivo **(escenarios).**
- El **conjunto completo** de casos de uso especifica todas las posibles formas de usar el sistema.

Actor

- Un **actor** representa un conjunto coherente de roles que juegan los usuarios de los casos de uso al interaccionar con el sistema.
- **Roles** desempeñados por personas, dispositivos u otros sistemas.
- El **tiempo** puede ser un actor (procesos iniciados automáticamente por el sistema).
- **No forman parte del sistema.**

Modelo de casos de uso

Actor

- Un usuario puede jugar diferentes roles.
- En la realización de un caso de uso pueden intervenir diferentes actores.
- Un actor puede intervenir en varios casos de uso.
- Identificar casos de uso mediante actores y eventos externos.
- Un actor necesita el caso de uso y/o participa en él.

Tipos de actores

- **Principal:** Solicita al sistema el cumplimiento de un objetivo.
- **Secundarios:** El sistema necesita de ellos para satisfacer un objetivo.

Escenario

- Un caso de uso describe un conjunto de secuencias de interacciones entre los actores principales y el sistema: escenarios.
- Un escenario es una historia particular de uso de un sistema (instancia de un caso de uso).
- Tipos de escenarios: principal (éxito) y alternativos (éxito, error y excepción).

Descripción de un caso de uso

- **Documentos de texto, no diagramas.**
- **Debe ser legible y comprensible para un usuario no experto.**
- **Debe especificar principalmente los actores y los escenarios (principal y alternativos).**

Formatos

- **Breve: resumen conciso (un párrafo) de los escenarios.**
- **Informal: un párrafo informal para describir cada escenario.**
- **Completo: Se describe con detalle todos los pasos y variaciones (plantilla).**
- **Notaciones gráficas: Diagramas de Secuencia de Sistema.**

Plantilla descripción de casos de uso (formato completo)

Caso de Uso: Identificador.

Descripción: Objetivo a conseguir.

Actores: Lista de actores.

Precondiciones: Condiciones que deben cumplirse para que se realice el caso de uso.

Postcondiciones: Condiciones que deben cumplirse una vez realizado el caso de uso.

Escenario principal: Interacciones entre los actores y el sistema necesarias para obtener el objetivo.

Extensiones: Escenarios alternativos.

Variaciones: cualquier variación en los pasos (opcional).

No-funcional: lista requisitos no funcionales (opcional).

Cuestiones: Lista de cuestiones que permanecen por resolver (opcional).

Ejemplo descripción caso de uso

Caso de uso: validar usuario (contexto cajero automático) (formato breve)

“El sistema pide el NIP. El cliente lo introduce y pulsa Enter. El sistema comprueba la validez del NIP. Si es válido el sistema acepta la entrada y finaliza el caso de uso. En caso contrario se reinicia el caso de uso. Si el NIP introducido es inválido tres veces consecutivas, el sistema cancela la transacción completa y se queda con la tarjeta. El cliente puede cancelar la transacción en cualquier momento y se reinicia el caso de uso”.

¿Sabes identificar y clasificar los escenarios del caso de uso?

CU: validar usuario (formato informal)

Flujo Principal: **“El sistema pide el NIP. El cliente lo introduce a través del teclado y pulsa Enter. El sistema comprueba la validez del NIP. Si es válido el sistema acepta la entrada y finaliza el caso de uso”.**

Flujo Excepcional: **“El cliente puede cancelar la transacción en cualquier momento, pulsando el botón Cancelar, reiniciando el caso de uso”.**

Flujo Excepcional: **“Si el NIP introducido es inválido entonces se reinicia el caso de uso. Si esto ocurre tres veces, el sistema cancela la transacción completa y se queda con la tarjeta”.**

CU: validar usuario (formato completo)

Caso de Uso: Validar usuario.

Descripción: Validar el NIP que el cliente introduce en el cajero automático.

Actores: Cliente.

Precondiciones: El cajero automático está a la espera de que un cliente introduzca su NIP.

Postcondiciones: El cajero acepta la conexión del cliente y muestra la pantalla con las operaciones disponibles.

Escenario principal:

1. El *Sistema* solicita que el cliente introduzca su NIP.
2. El *Cliente* introduce su NIP y pulsa Enter.
3. El *Sistema* comprueba que el NIP introducido es válido y acepta la conexión del cliente.

Extensiones (Flujos alternativos):

3a. El NIP introducido es inválido (primera/segunda vez):
   1. El Sistema muestra el error y rechaza la conexión (paso 2)

3b. El NIP introducido es inválido por tercera vez consecutiva:
   1. El Sistema muestra el error NIP inválido.
   2. El Sistema cancela la transacción completa y se queda con la tarjeta.

*a. En cualquier momento el cliente cancela la transacción
   1. El *Sistema* reinicia el caso de uso.

Modelo de casos de uso

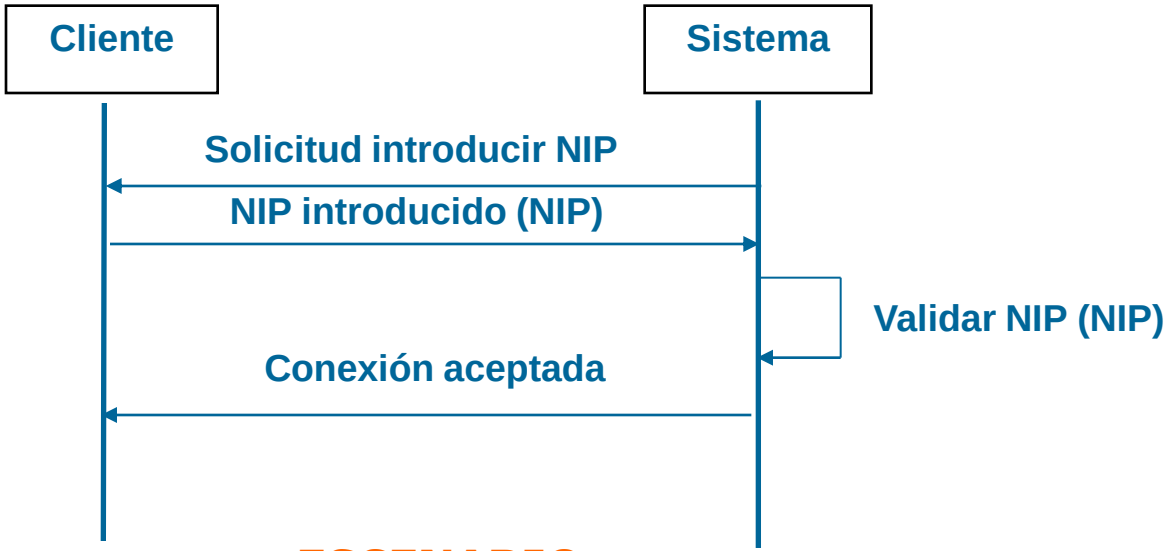
Variaciones: Ninguna.

No-funcional:

- Interfaz de usuario con pantalla táctil en un monitor de pantalla plana.
- Tiempo de respuesta para la autorización de la conexión de 5 segundos el 90 % de las veces.

Cuestiones: Ninguna.

CU: validar usuario (diagrama de secuencia de sistema)



ESCENARIO Principal

CU: Realizar venta (plantilla)

Caso de Uso: Realizar venta.

Descripción: Un cliente llega al TPV con un conjunto de artículos. El cajero registra los artículos y se genera un ticket. El cliente paga en efectivo y recoge los artículos.

Actores: Cajero (principal), Cliente (secundario).

Precondiciones: El cajero se identifica.

Postcondiciones: Se registra la venta y el pago, se actualiza el inventario y se genera el recibo para el cliente.

CU: Realizar venta (plantilla)

Escenario principal:

1. El *Cliente* llega al TPV con los artículos.
2. El *Cajero* inicia una nueva venta.
3. El *Cajero* introduce el identificador de un artículo.
4. El *Sistema* comprueba que el artículo es correcto, registra la línea de venta y presenta la descripción del artículo, el precio y la suma parcial de la venta.

El Cajero repite los pasos 3 y 4 hasta que finalice la venta. 

5. El *Sistema* presenta el total de la venta con los impuestos incluídos.
6. El *Cajero* le dice al Cliente el total a pagar.
7. El *Cliente* paga y el sistema gestiona el pago.
8. El *Sistema* registra la venta completa y actualiza el inventario.
9. El *Sistema* presenta el recibo.

CU: Realizar venta (plantilla)

Extensiones (Flujos Alternativos):

4a. Identificador de artículo no válido.

1. El *Sistema* comprueba que el identificador del artículo no es válido, señala el error y rechaza el artículo (paso 3)

3-4a. El *Cliente* solicita eliminar un artículo de la compra.

1. El *Cajero* selecciona el artículo a eliminar.
2. El *Sistema* elimina el artículo de la venta y actualiza la suma parcial.

...

7a. Pago en efectivo

1. El *Cajero* introduce la cantidad entregada por el cliente.
2. El *Sistema* muestra cantidad a devolver.
3. ...

....

CU: Realizar venta (plantilla)

Variaciones:

- El identificador podría ser cualquier esquema de código.

No funcional:

- Interfaz de usuario con pantalla táctil en un monitor de pantalla plana. El texto debe ser visible a un metro de distancia.
- Tiempo de respuesta para autorización de crédito de 30 sg. El 90% de las veces.
- La entrada de información de la tarjeta se realiza mediante un lector de tarjetas.

Cuestiones Pendientes:

- Explorar cuestiones de recuperación de accesos a servicios remotos.
- ¿Qué adaptaciones son necesarias para diferentes negocios?

Modelo de casos de uso

Analiza la descripción del caso de uso “Realizar venta”

- Identifica y clasifica los Escenarios
- Modifica la descripción del caso de uso considerando:

- El Cajero introduce el identificador del artículo y el número de unidades.
- El Sistema comprueba que el stock del artículo es >= número de unidades introducidas.
- Añadir la opción de pago de la compra con Paypal.
- El Cliente selecciona pago con Paypal y el Cajero introduce los datos.
- El Sistema registra el cargo de la compra.

Identificación y clasificación de los escenarios

Escenario principal (éxito): Compra Ok (pago con tarjeta).

Escenario alternativo (éxito): Compra OK (con eliminación de artículos).

Escenario de error: Artículo no existe.

Modificación de la descripción del caso de uso

Escenario principal:

3. El *Cliente* introduce el identificador de un artículo de la compra y el número de unidades.
4. El *Sistema* comprueba que el artículo es correcto y que hay stock suficiente (>= número unidades), registra la línea de compra y presenta la descripción del artículo, el precio y la suma parcial de la compra.

El Cajero repite los pasos 3 y 4 hasta que finalice la venta.

5. El *Sistema* presenta el total de la compra con los impuestos incluidos.
6. El *Sistema* solicita al Cliente que realice el pago de la compra.
7. El *Cliente* paga y el Sistema gestiona el pago ….

Extensiones (escenarios alternativos) :

4b. Stock del artículo insuficiente.
   1. El *Sistema* comprueba que el stock es menor que el número de unidades solicitadas, indica el error y rechaza el artículo (paso 3).

7b. El Cliente paga con Paypal.
   1. El *Cliente* selecciona pago con Paypal y el Cajero introduce los datos del pago.
   2. El *Sistema* registra el cargo de la compra.

¿Cuál es el mejor formato?

- No existe un mejor formato (diferentes preferencias, complejidad del caso de uso).
- Las secciones se pueden añadir y quitar.
- Los nombres de los títulos se pueden cambiar (no es especialmente importante).
- Clave: Describir los detalles del escenario principal de éxito y los escenarios alternativos de alguna forma.

Relaciones entre casos de uso

Generalización

- Un cdu hereda el comportamiento y significado de otro.

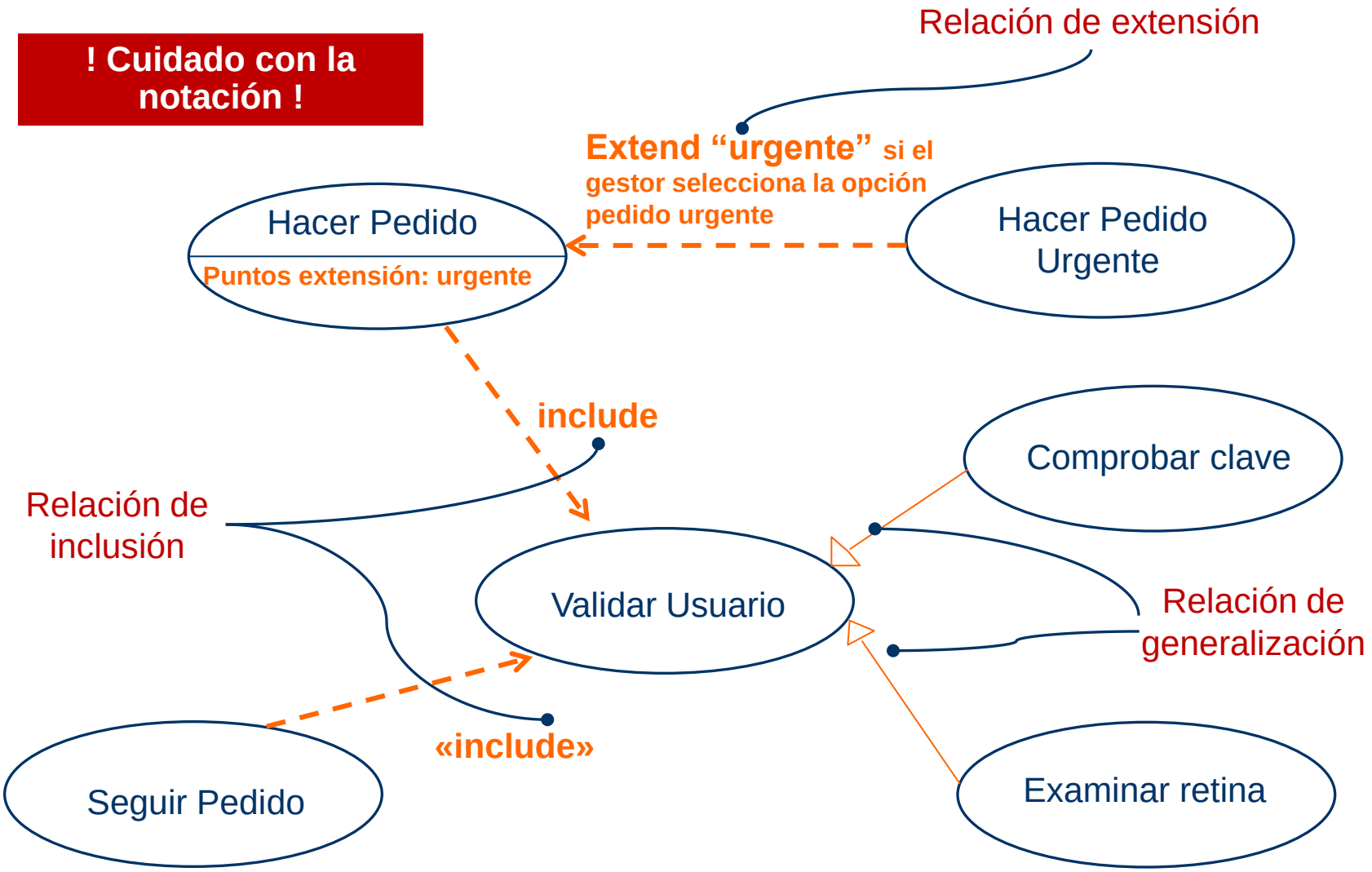
Inclusión

- Un cdu base incorpora explícitamente el comportamiento de otro en algún lugar de su secuencia.

Extensión

- Un cdu base incorpora implícitamente el comportamiento de otro cdu en el lugar especificado en el cdu base.

Relaciones entre casos de uso



Relación de inclusión

Un cdu base incorpora **explícitamente** el comportamiento de otro en algún lugar de su secuencia.

Permite **factorizar un comportamiento en un caso de uso aparte** y evitar repetirlo en diferentes casos de uso.

Ejemplo

CU: Hacer pedido

Escenario principal:

1. El *Sistema* solicita que se introduzca el identificador del usuario
2. El *Gestor* introduce su identificador
3. Include (Validar usuario)
4. El *Sistema* solicita el número del pedido.
5. El *Gestor* introduce el número de un pedido nuevo.
6. …

CU: Seguir pedido

Escenario principal:

1. El *Sistema* solicita que se introduzca el identificador del usuario
2. El *Gestor* introduce su identificador
3. Include (Validar usuario)
4. El *Sistema* solicita el número del pedido.
5. El *Gestor* introduce el número del pedido del que quiere hacer el seguimiento
6. ….

Relación de extensión

Un cdu base incorpora implícitamente el comportamiento de otro cdu en el lugar especificado en el cdu base (punto de extensión).

Sirve para modelar:

- la parte opcional del sistema.
- un subflujo que sólo se ejecuta bajo ciertas condiciones.
- varios flujos que se pueden insertar en un punto.

Relación de extensión

Ejemplo

CU: Hacer pedido

Nivel: Función

Puntos de extensión: “urgente” en 9a.

Escenario principal:

1. El *Sistema* solicita el identificador del usuario.
2. El usuario introduce su identificador.
3. Include (Validar usuario).
4. El *Sistema* solicita el número del pedido.
5. El Gestor introduce el número de un pedido nuevo.
6. El Sistema comprueba que el pedido es válido y solicita los datos del pedido.
7. El Gestor introduce los datos del pedido.
8. El Sistema comprueba que los datos del pedido son correctos.
9. El Pedido es normal y el Sistema registra los datos del pedido.
10. …

Extensiones:

9a. Pedido urgente.

Forma de obtener los casos de uso

1) Identificar los usuarios del sistema.

2) Encontrar todos los roles que juegan los usuarios y que son relevantes para el sistema

3) Para cada role identificar todas las formas de interactuar con el sistema (objetivos)

4) Crear un caso de uso por cada objetivo

5) Estructurar los casos de uso (!Cuidado!)

6) Revisar y validar con el usuario

Recomendaciones

- **Representar un objetivo de subfunción como caso de uso si la subfunción se repite o es precondición en muchos casos de uso de nivel de objetivos de usuario. Ej. *Identificar y validar usuario*.**

- **Excepción caso de uso por objetivo: Agrupar objetivos separados CRUD (crear, recuperar, actualizar, eliminar) en un caso de uso llamado por convención *Gestionar <X>*.**

- **Error común: Definir muchos casos de uso a un nivel muy bajo. Ej. *eliminar una línea de pedido***

- **Error común: no identificar como casos de uso las tareas que inicia el propio sistema. Ej. *anular reservas pasados quince días.***

Recomendaciones

- Especificar casos de uso no es una actividad de dibujar diagramas sino de escribir con el detalle necesario el flujo principal y los flujos alternativos: centrado en la escritura en vez del dibujo.
- El objetivo inicial es identificar los actores principales y, a partir de sus objetivos, encontrar los casos de uso.
- Los actores principales deben interactuar con el sistema.
- Un Caso de Uso NO debe considerar cuestiones de implementación.

Recomendaciones

- **No es necesario aplicar la abstracción**

  ¡Usa casos de uso Concretos!

- **Evita redes complicadas de casos de uso: Cuidado con las relaciones *include* y *extend.***

- **Los casos de uso sólo consideran los requisitos funcionales del proyecto, hay que añadir los no-funcionales.**

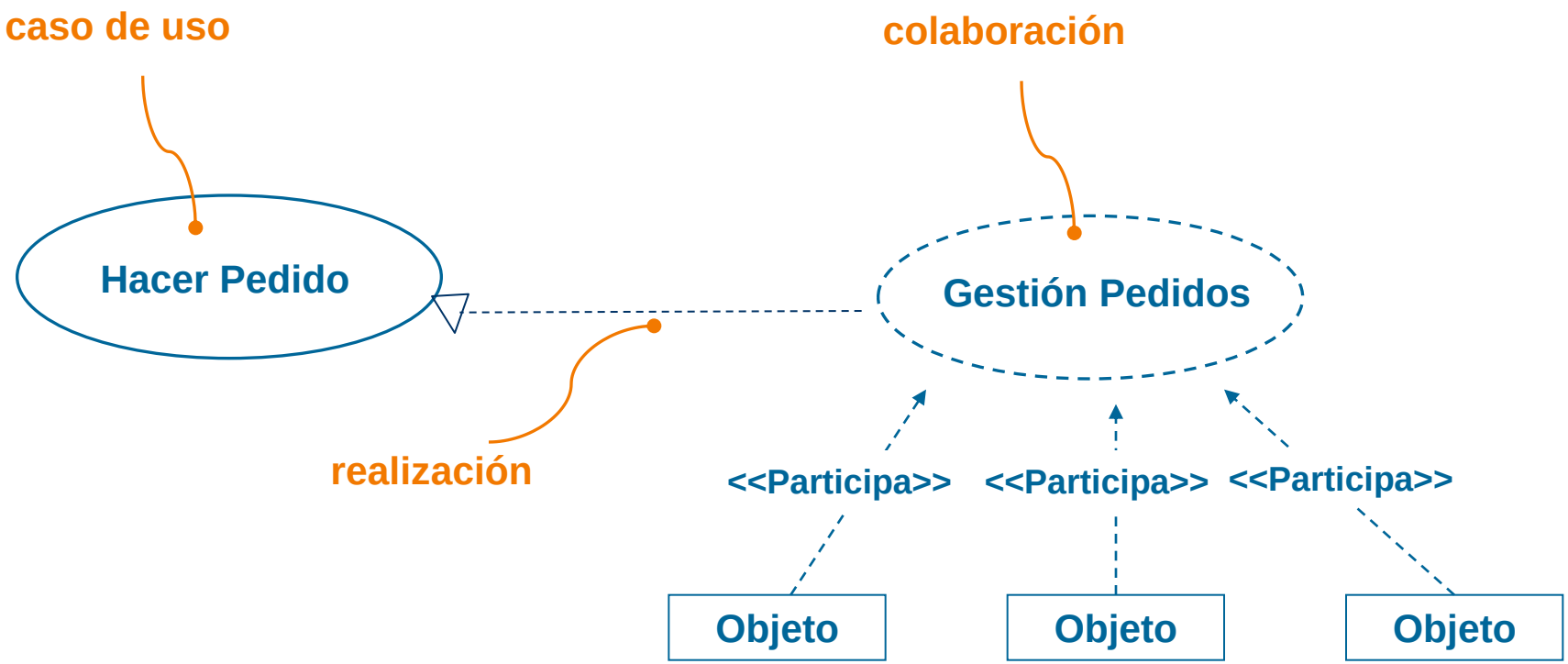
Casos de uso y colaboraciones

- Un caso de uso describe un comportamiento esperado del sistema, pero no especifica cómo se implementa.
- Un caso de uso se implementa a través de una colaboración:

  "Sociedad de clases y otros elementos que colaborarán para realizar el comportamiento expresado en un caso de uso"

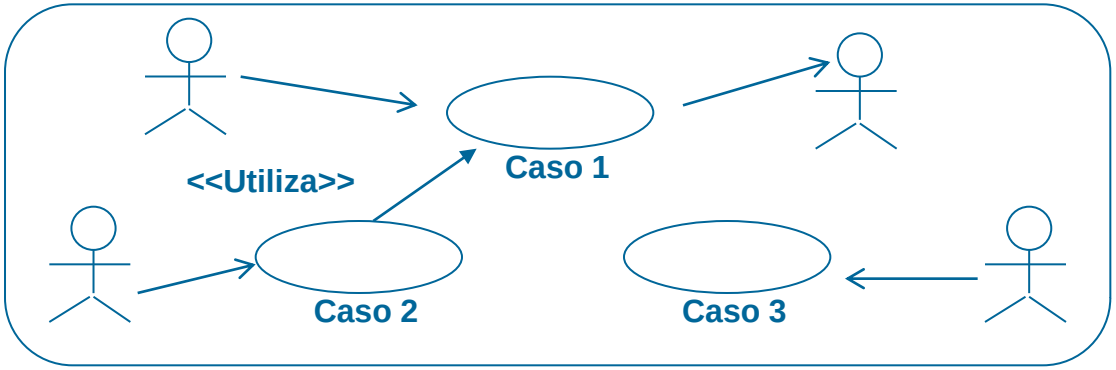
- Una colaboración tiene una parte estática (diagramas de clases) y una parte dinámica (diagramas de secuencia).

Casos de uso y colaboraciones

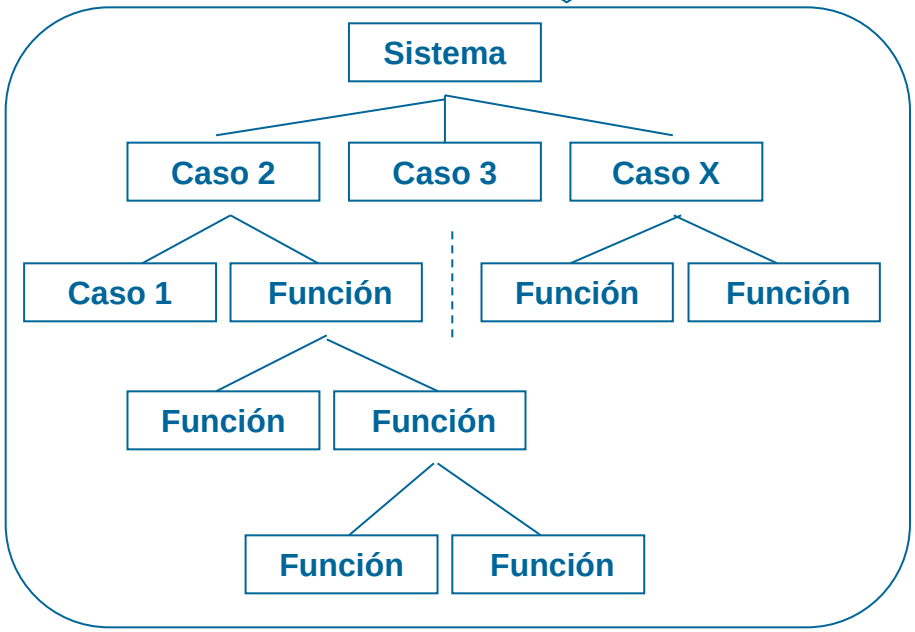


Modelo de casos de uso

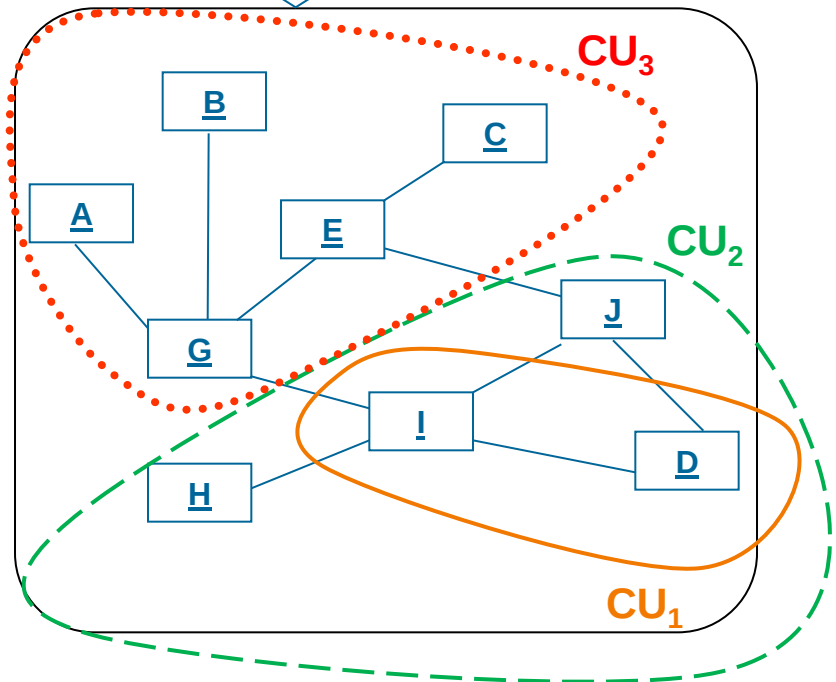
Casos de uso y colaboraciones



Descomposición estructurada



Descomposición de objetos



Casos de uso y colaboraciones

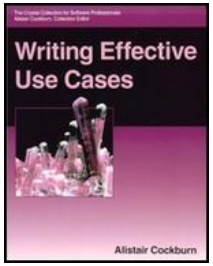
“El objetivo de la arquitectura del sistema es encontrar el conjunto mínimo de colaboraciones bien estructuradas que satisfacen el comportamiento especificado en todos los casos de uso del sistema”

Bibliografía

- Booch, G.; Jacobson, J.; Rumbaugh, J.M. El lenguaje unificado de modelado. Manual de Referencia, 2ª ed. Ed. Addison Wesley, 2007.

- Booch, G.; Jacobson, J.; Rumbaugh, J.M. El lenguaje unificado de modelado. Guía de Usuario, 2ª ed. Ed. Addison Wesley, 2007.

- A. Cockburn. Writing effective use cases. Addison-Wesley, 2000.

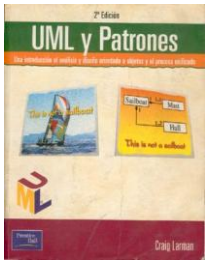


Bibliografía

- http://www.usecases.org
- http://alistair.cockburn.us/usecases/usecases.html

- IEEE Std. 830. Recommended Practice for Software Requirements Specifications.

- C. Larman. UML y Patrones: Una introducción al análisis y diseño orientado a objetos y al proceso unificado. 2ª ed. Prentice-Hall, 2003.



- Pressman, R. Ingeniería del Software. Un enfoque práctico, 6ª ed. McGraw Hill, 2005.

Bibliografía

- **Sanchez, S.; Sicilia, M.A.; Rodríguez D. Ingeniería del Software. Un enfoque desde la guía SWEBOK. Ed. Garceta, 2011.**

- **Sommervile, I. Ingeniería del Software, 8ª ed. Pearson Education, 2007.**