INGENIERÍA DE SOFTWARE I

Tema 7 - Análisis Orientado a Objetos

Grado en Ingeniería Informática
Fecha de última modificación: 9-2-2025

Dr. Francisco José García-Peñalvo / fgarcia@usal.es
Dra. Alicia García-Holgado / aliciagh@usal.es
Dra. Andrea Vázquez-Ingelmo / andreavazquez@usal.es

Departamento de Informática y Automática
Universidad de Salamanca

Resumen

Resumen	El análisis orientado a objetos consiste en una serie de técnicas y actividades mediante las que los requisitos identificados en la fase de elicitación son analizados, refinados y estructurados. El objetivo es una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que ayude a estructurar el sistema. El resultado consistirá en un modelo del sistema, modelo objeto, que describa el dominio del problema y que deberá ser correcto, completo, consistente y verificable
Descriptores	Análisis orientado a objetos; modelo de dominio; clase conceptual; Proceso Unificado; objeto de entidad; objeto de interfaz; objeto de control
Bibliografía	[Booch et al., 2007] Capítulos 8, 9 y 10 [Jacobson et al., 2000] Capítulo 8 [Larman, 2003] Capítulos 9, 10, 11, 12, 26 y 27 [Sommerville, 2011] Capítulo 5

Esquema

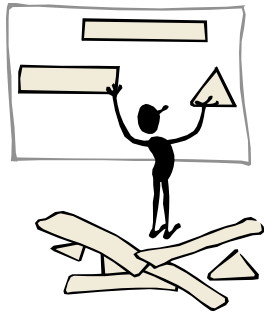
- Introducción
- Análisis orientado a objetos
- Modelo del dominio
- Análisis en el Proceso Unificado
- Aportaciones principales del tema
- Ejercicios
- Lecturas complementarias
- Referencias

https://unsplash.com/photos/ij5_qCBpIVY

1. Introducción

¿De dónde se parte?

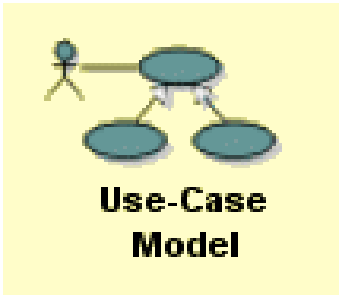
Usuarios, clientes...



Desarrolladores

Requisitos

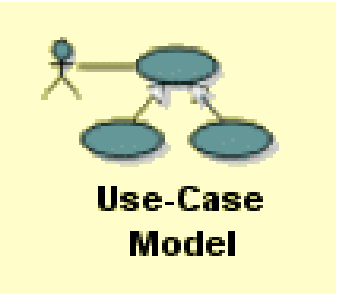
Escenarios / casos de uso



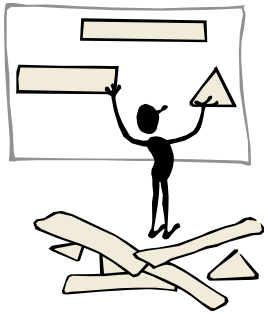
¿Dónde se quiere llegar?

Usuarios, clientes...

Escenarios / casos de uso



Requisitos



Desarrolladores

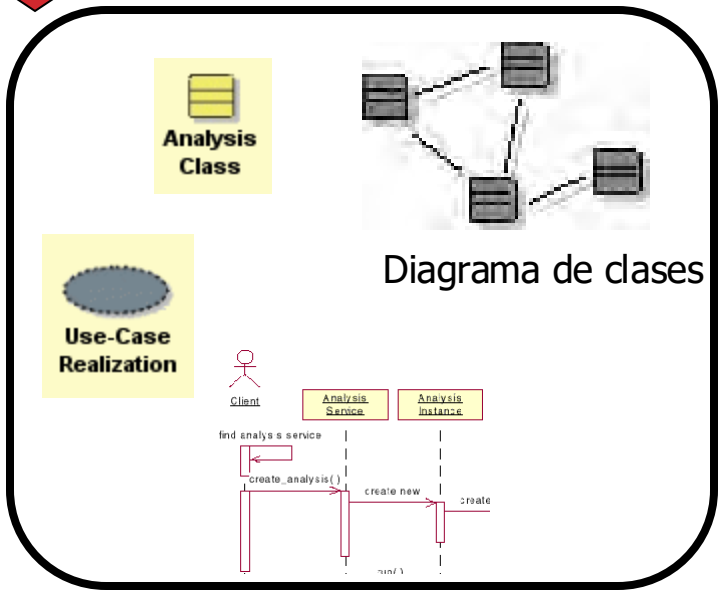
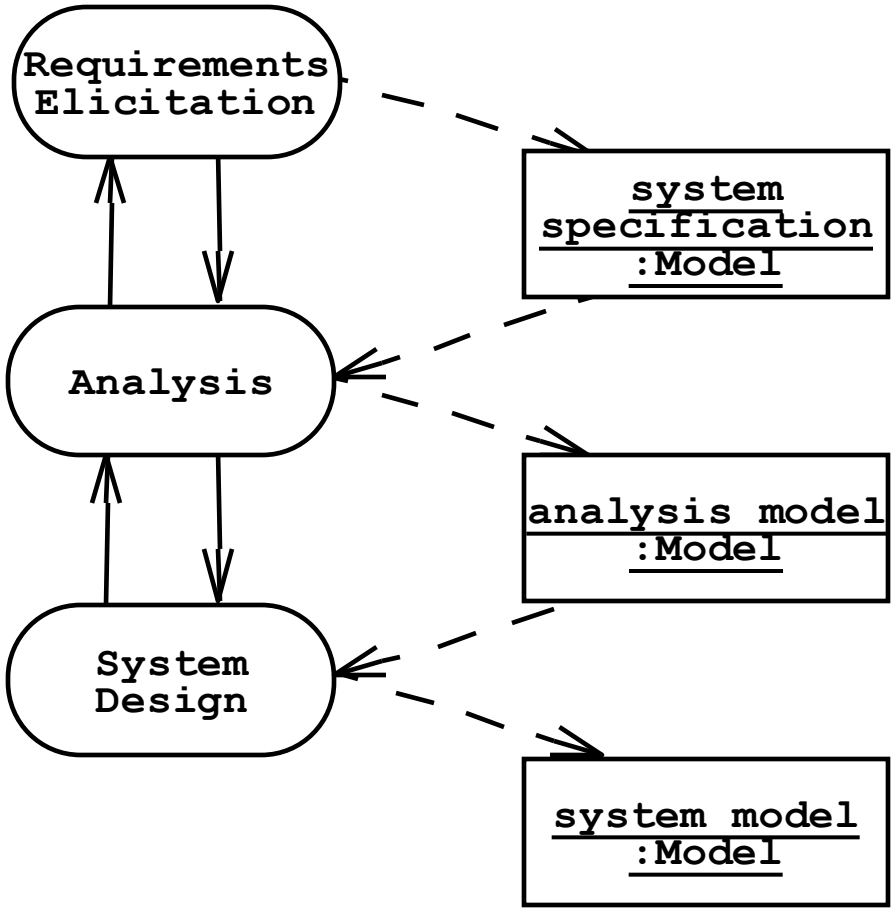


Diagrama de clases

Productos de la recogida y análisis de requisitos



[Bruegge y Dutoit, 2000]

Definición de análisis

- En términos generales se define análisis como "la distinción y separación de las partes de un todo hasta llegar a conocer sus principios o elementos" [RAE, 2024]
- Desde un punto de vista informático se define análisis como "el estudio, mediante técnicas informáticas, de los límites, características y posibles soluciones de un problema al que se aplica un tratamiento por ordenador" [RAE, 2024]
- Para la Ingeniería del *Software* el análisis es la parte del proceso de desarrollo de *software* cuyo propósito principal es realizar un modelo del dominio del problema
  - Se puede definir más precisamente como "el proceso del estudio de las necesidades de los usuarios para llegar a una definición de los requisitos del sistema, de *hardware* o de *software*, así como el proceso de estudio y refinamiento de dichos requisitos" [IEEE, 1999]

El objeto del análisis orientado a objetos

- Analizar los requisitos en la forma de un modelo de análisis es importante por varios motivos [Jacobson et al., 1999]
  - Un modelo de análisis ofrece una especificación más precisa de los requisitos que la que se tiene como resultado de la captura de requisitos, incluyendo al modelo de casos de uso
  - Un modelo de análisis se describe utilizando el lenguaje de los desarrolladores y puede, por tanto, introducir un mayor formalismo y ser utilizado para razonar sobre los funcionamientos internos del sistema
  - Un modelo de análisis estructura los requisitos de un modo que facilita su comprensión, su preparación, su modificación y, en general, su mantenimiento
  - Un modelo de análisis puede considerarse como una primera aproximación al modelo de diseño y es, por tanto, una entrada fundamental cuando se da forma al sistema en el diseño y en la implementación

2. Análisis orientado a objetos

Definición

El análisis orientado a objetos es el proceso que modela el dominio del problema mediante la identificación y la especificación de un conjunto de objetos semánticos que interaccionan y se comportan de acuerdo a los requisitos del sistema

[Monarchi y Puhr, 1992]

- Permite describir el sistema en los mismos términos que el mundo real
- Se centra en la comprensión del espacio (dominio) del problema
- Contiene elementos de síntesis
  - La abstracción de requisitos de usuario y la identificación de los objetos clave del dominio es seguida del ensamblaje de estos objetos en estructuras de forma que soporten el diseño en fases posteriores

Generalidades (i)

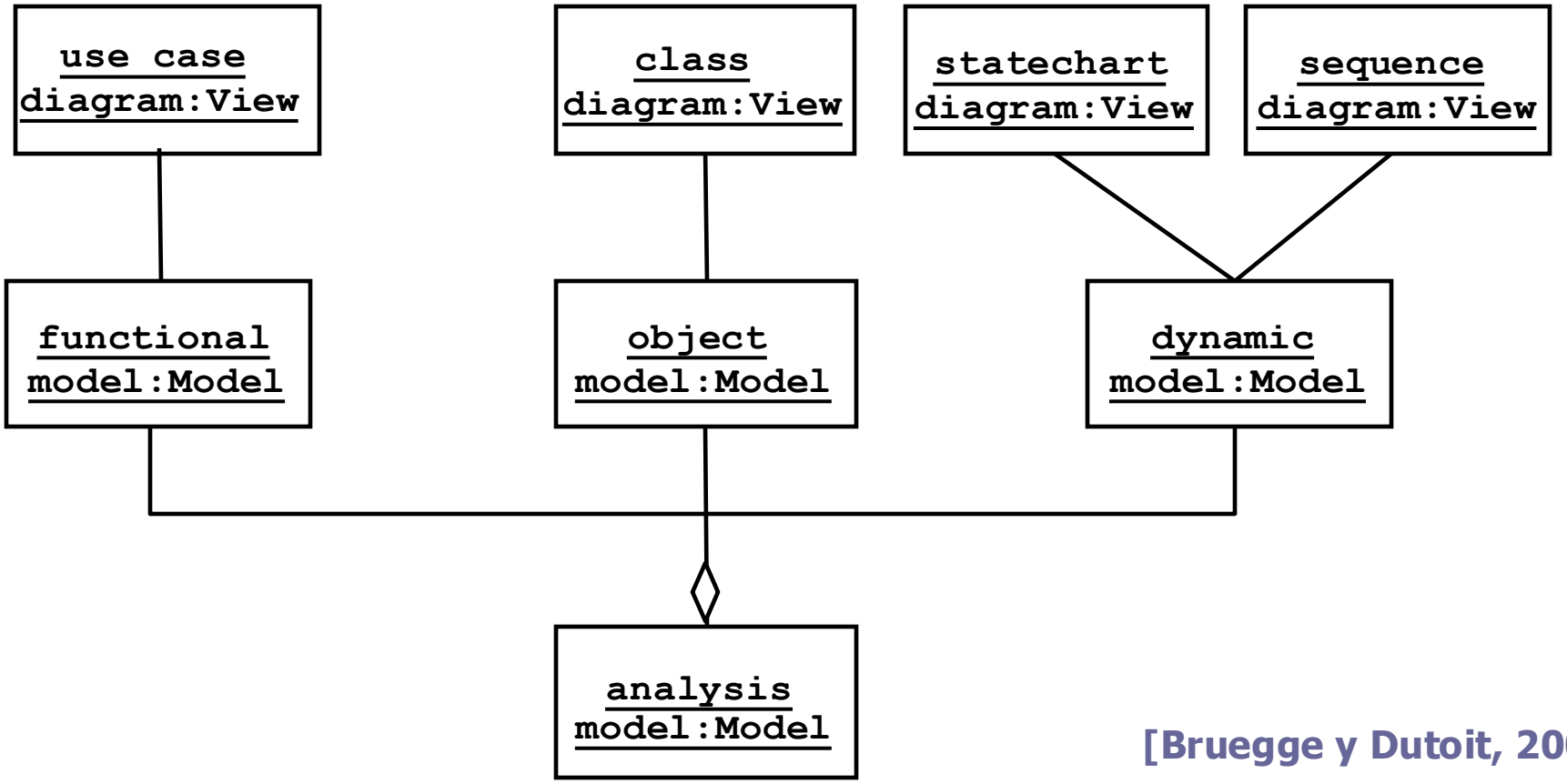
- Difícil determinar dónde acaba el análisis orientado a objetos y dónde comienza el diseño orientado a objetos
- El objetivo es modelar la semántica del problema en términos de objetos distintos pero relacionados
- El análisis casa con el dominio del problema
- Los objetos del dominio del problema representan cosas o conceptos utilizados para describir el problema (**objetos semánticos**)
- Los objetos del dominio del problema tienen una equivalencia directa en el entorno de la aplicación
- Se centra en la representación del problema
  - Identificar abstracciones que contengan el significado de las especificaciones y de los requisitos

Generalidades (ii)

- El modelo de casos de uso identifica secuencias de eventos e interacciones entre actores y el sistema
- El **modelo de análisis** especifica las clases de objetos que se encuentran o existen en el sistema
- No existen reglas fijas para esta transformación
- Se centra en la elaboración de un modelo del sistema, el modelo de análisis
  - Modelo funcional
    - Representado por los casos de uso
  - Modelo objeto análisis
    - Representado por los diagramas de clase y objetos
  - Modelo dinámico
    - Representado por los diagramas de secuencia y los diagramas de transición de estados

Estructura del modelo de análisis

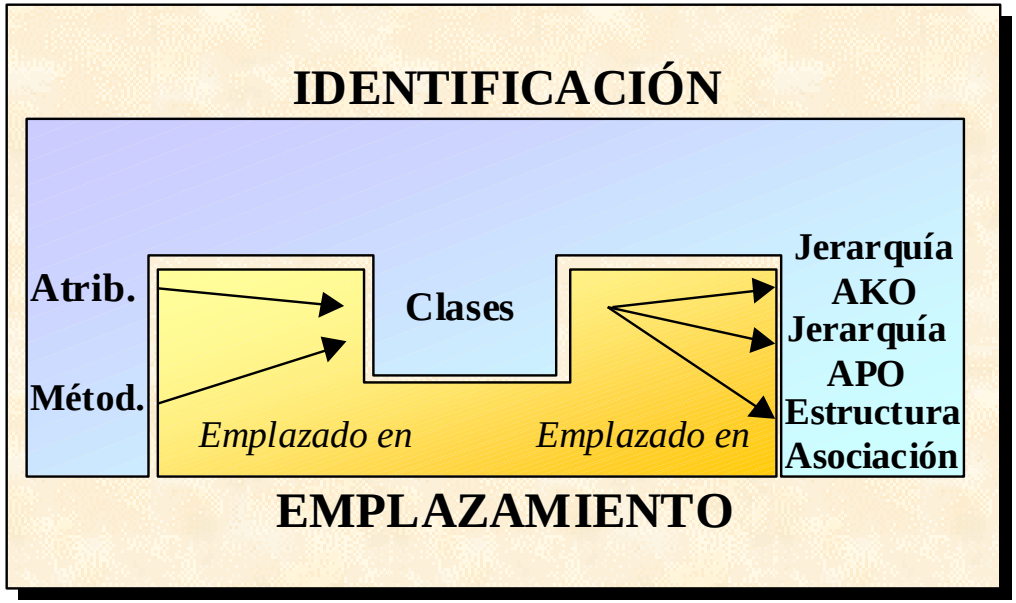
- El **Modelo de Análisis** estructura el sistema independientemente del entorno actual de implementación



[Bruegge y Dutoit, 2000]

Actividades del análisis orientado a objetos

- La identificación de las clases semánticas, los atributos, el comportamiento y las relaciones (generalizaciones, agregaciones y asociaciones)
- El emplazamiento de las clases, atributos y comportamiento
- La especificación del comportamiento dinámico mediante paso de mensajes



[Monarchi y Puhr, 1992]

Tipos de proceso en análisis

- Existen diferentes enfoques de proceso en el análisis
  - Centran en la información (datos) del sistema
  - Centran en la funcionalidad (comportamiento) del sistema
  - Síntesis de los dos procesos anteriores
- El Proceso Unificado [Jacobson et al., 1999] sigue el enfoque de síntesis
  - Inicio por la funcionalidad (**Casos de uso**)
  - Refinamiento por la información (**Diagramas de Clases**)
  - Consolidación por la funcionalidad (**Diagramas de secuencia /colaboración**)

https://bit.ly/38jYrY2

3. Modelo del dominio

Introducción

- Su utilidad radica en ser una forma de "inspiración" para el diseño de los objetos *software*
- Es entrada para muchos de los artefactos que se construyen en un proceso *software*
- Un modelo de dominio muestra las **clases conceptuales** significativas en un dominio del problema
  - Se centra en las abstracciones relevantes, vocabulario del dominio e información del dominio
- Es el artefacto clave del análisis orientado a objetos
- En UML se utilizan los diagramas de clases para representar los modelos de dominio

Un modelo de dominio es una representación de las clases conceptuales del mundo real, no de componentes *software*. No se trata de un conjunto de diagramas que describen clases *software*, u objetos *software* con responsabilidades

[Larman, 2002]

Guías para hacer un modelo de dominio

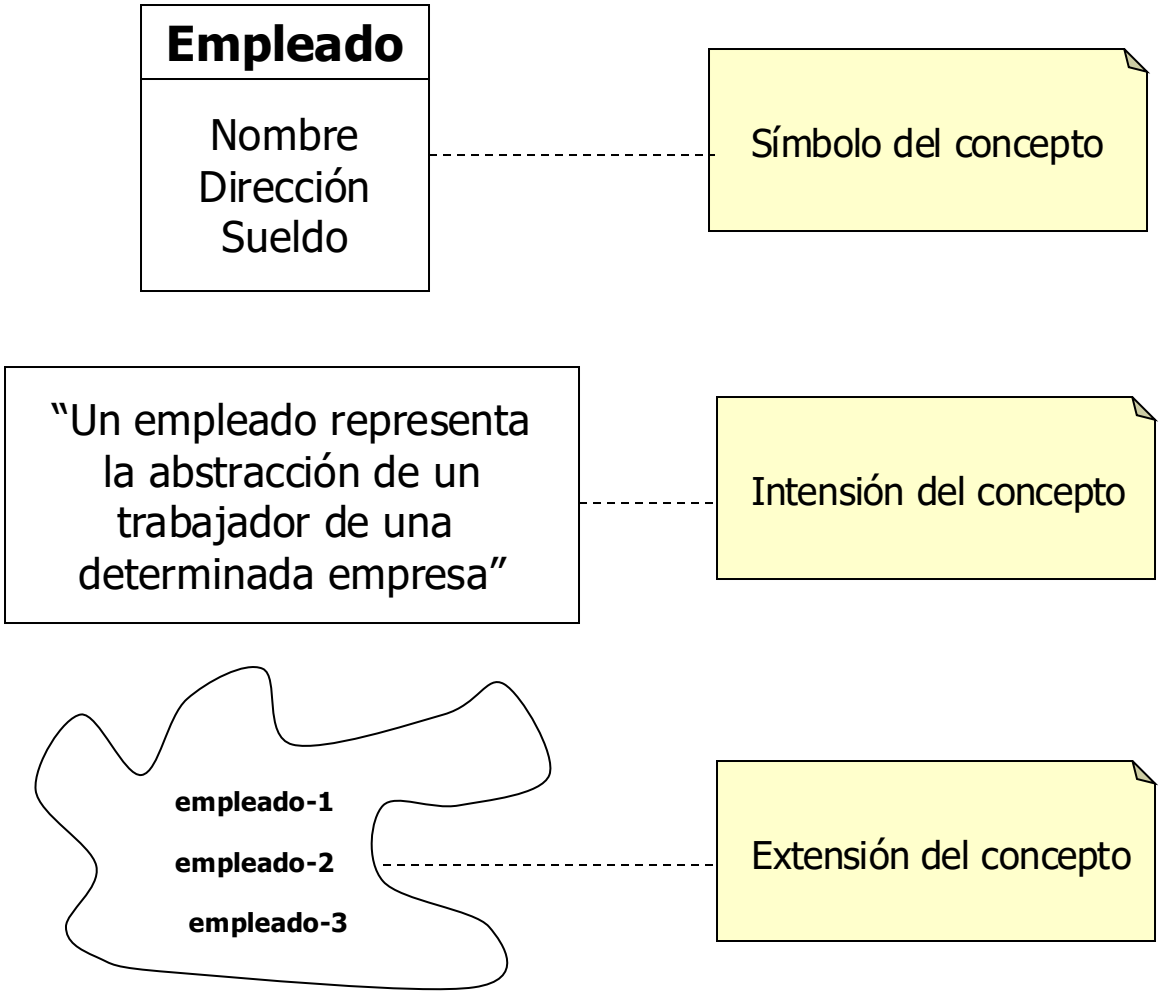
- Listar las clases conceptuales candidatas relacionadas con los requisitos actuales en estudio
- Representar las clases en un modelo de dominio
- Añadir las asociaciones necesarias para registrar las relaciones que hay que mantener en memoria
- Añadir los atributos necesarios para satisfacer los requisitos de información

[Larman, 2002]

Identificación de clases conceptuales (i)

- El modelo de dominio muestra las clases conceptuales o vocabulario del dominio
- Informalmente una clase conceptual es una idea, cosa u objeto
- Formalmente, una clase conceptual puede considerarse en términos de su símbolo, intensión y extensión [Martin y Odell, 1995]
  - **Símbolo:** palabras o imágenes que representan una clase conceptual
  - **Intensión:** la definición de una clase conceptual
  - **Extensión:** el conjunto de ejemplos a los que se aplica la clase conceptual

Identificación de clases conceptuales (ii)



Identificación de clases conceptuales (iii)

- Objetivo crear un modelo de dominio de clases conceptuales significativas del dominio de interés
- Se van añadiendo clases al modelo del dominio a medida que se revisan los escenarios identificados en los casos de uso
- Es mejor especificar en exceso un modelo de dominio con muchas clases conceptuales de grano fino que especificar por defecto [Larman, 2002]
  - El modelo no es mejor por tener pocas clases conceptuales
  - Es normal obviar clases conceptuales durante la identificación inicial para descubrirlas más tarde (incluso en diseño) al considerar atributos y asociaciones
  - No se deben excluir clases conceptuales porque los requisitos no indican necesidad obvia de registrar información sobre ella o porque la clase conceptual no tenga atributos
- Estrategias para identificar clases conceptuales
  - Utilizar una lista de categorías de clases conceptuales
  - Identificar frases nominales
  - Patrones de análisis [Fowler, 1996; Hay, 1996]
    - Un patrón de análisis es un modelo de dominio parcial y existente que ha sido creado por expertos

Identificación de clases conceptuales (iv)

Método: Lista de categorías [Shlaer y Mellor, 1988; Larman, 2002]

- Objetos tangibles o físicos
  - Avión, asiento, billete, equipaje, tarjeta de embarque…
- Roles: papeles desempeñados por personas u organizaciones
  - Piloto, agente de ventas, pasajero…
- Incidentes: representan la ocurrencia de un evento, algo que sucede en un momento determinado
  - Cancelación de vuelo, vuelo, aterrizaje, colisión…
- Interacciones: transacciones o contratos que se dan entre dos o más objetos del modelo
  - Reserva, venta de billete, pago…
- Líneas de las transacciones
  - Línea de venta…
- Especificaciones: propias de aplicaciones de inventario o fabricación. Relacionados con los estándares o definiciones de elementos. Descripciones
  - Descripción del vuelo…
- Organizaciones
  - Departamento de ventas, compañía aérea…
- Lugares
  - Tienda…
- Contenedores
  - Avión, tienda, lata…
- Cosas contenidas
  - Artículo, pasajero
- Conceptos abstractos
  - Ansia, claustrofobia…
- Otros sistemas informáticos externos al sistema
  - Control de tráfico aéreo, sistema de autorización de pago a crédito…

Identificación de clases conceptuales (v)

Método [Coad y Yourdon, 1990]. En primer lugar se buscan candidatos entre las siguientes categorías

- Otros sistemas: Sistemas externos que interaccionan con el sistema en estudio
  - Control de tráfico aéreo
- Dispositivos: Dispositivos físicos que interaccionan con el sistema en estudio intercambiando información control y datos. **No incluir** componentes de ordenador (discos, pantallas...)
  - Sensor
- Hechos (eventos a registrar): Equivalente a los incidentes de [Shlaer y Mellor, 1988]
- Roles
- Localizaciones: ¿De qué localizaciones físicas, oficinas o sitios se ha de tener conocimiento?
- Unidades organizativas
  - Compañía aérea

Identificación de clases conceptuales (vi)

Método [Coad y Yourdon, 1990]. De los candidatos se incluyen en el modelo aquéllas que cumplan una o más de las siguientes propiedades

- Guardar información: Se necesita guardar información acerca de las clases potenciales
  - Usuario del sistema
- Necesidad de servicio: Incorporan un conjunto de operaciones que pueden proveer servicios a otras clases
  - Partida: Proporciona información que caracteriza el estado de un juego – puntuación de los jugadores, tiempo de pensar…
- Atributos múltiples: La clase tiene más de un atributo
  - Balance: Representa una cantidad, esto es, balance como atributo de Cuenta
- Atributos comunes: Todas las instancias del "nombre" comparten los mismos atributos
  - Cliente (nombre, dirección, teléfono...)
- Operaciones comunes: Todas las operaciones definidas para el "nombre" se aplican al resto de las instancias del nombre
  - Cliente [getName()]
- Requisitos esenciales: Entidades externas conocidas por el sistema y que producen o consumen información

Identificación de clases conceptuales (vii)

Método: Análisis del lenguaje natural [Abbott, 1983] (i)

- Análisis lingüístico de los documentos de recogida de requisitos
  - Especificaciones, **documentación de casos de uso**, glosario, formularios, descripciones técnicas del sistema, notas de entrevistas…
- Es un método útil por su simplicidad, pero no demasiado preciso
  - No existe una correspondencia automática y mecánica de nombres a clases
  - El lenguaje natural no es muy preciso
  - Depende del estilo "literario" del analista
- Identificar nombres y frases nominales en las descripciones textuales y considerarlos como clases conceptuales o atributos candidatos

Identificación de clases conceptuales (viii)

Método: Análisis del lenguaje natural [Abbott, 1983] (ii)

- Identificar candidatos
  - Identificar los nombres (conceptos) relevantes para el problema
  - Identificar sinónimos, acrónimos y términos especiales
  - Identificar polisemias
- Categorizar los nombres. Combinación de esta técnica con las anteriores
  - Objetos concretos: coche
  - Personas y sus roles: cliente, empleado
  - Información de acciones o hechos: transferencia bancaria
  - Sitios: sala de espera
  - Organizaciones: división de coches sin conductor
  - Interfaces con el sistema: lista de espera
  - Relaciones entre objetos: compra de billete, alquiler de coche
  - Información: artículo, tipo de congreso

Identificación de clases conceptuales (ix)

Método: Análisis del lenguaje natural [Abbott, 1983] (iii)

- Eliminar de la lista de candidatos aquellos nombres y conceptos que no representen clases conceptuales independientes
  - Aquéllos para los que no se puedan identificar ni atributos ni operaciones
  - Los relacionados con detalles de implementación del modelo
- Elegir nombres significativos
  - En singular
  - Tan concretos como sea posible
  - Ni polisemias, ni acrónimos
    - Cuidado con los sinónimos
  - Deben representar a todos los atributos y operaciones que se les puedan asociar

Identificación de clases conceptuales (x)

Método: Análisis del lenguaje natural [Abbott, 1983] (iv)

- Heurísticas

Parte de la oración	Componente del modelo de objetos	Ejemplo
Nombre propio	Instancia	Puzzle
Nombre común	Clase	Juguete
Verbo acción	Operación	Guardar
Verbo de existencia	Clasificación	Es un
Verbo de posesión	Composición	Tiene un
Verbo afirmativo	Condición de invarianza	Posee
Adjetivo	Valor o clase (atributo)	Inadecuado
Frase adjetiva	Asociación	Cliente con niños
	Operación	Cliente que compró el puzzle
Verbo transitivo	Operación	Entrar
Verbo intransitivo	Excepción o suceso	Depende

Identificación de asociaciones (i)

- Se deben de incluir las siguientes asociaciones en un modelo del dominio [Larman, 2002]
  - Asociaciones de las que es necesario conservar el conocimiento de la relación durante algún tiempo (asociaciones **necesito-conocer**)
  - Asociaciones derivadas de la lista de categorías comunes de asociaciones
- Se deben eliminar
  - Las relaciones no permanentes
  - Aquéllas que sean irrelevantes para la especificación
  - Orientadas a la implementación
  - Las que pueden derivarse a partir de otras asociaciones
- Se deben definir nombres de asociación, roles, multiplicidad
- Guía para las asociaciones [Larman, 2002]
  - Centrarse en las asociaciones **necesito-conocer**
  - Es más importante identificar clases conceptuales que identificar asociaciones
  - Demasiadas asociaciones tienden a confundir un modelo de dominio en lugar de aclararlo. Su descubrimiento puede llevar tiempo, con beneficio marginal
  - Evitar mostrar asociaciones redundantes o derivadas

Identificación de asociaciones (ii)

- Lista de asociaciones comunes [Larman, 2002] (i)
  - A es una parte física de B
    - Ala – Avión
  - A es una parte lógica de B
    - EtapaVuelo – RutaVuelo
  - A está contenido físicamente en B
    - Pasajero – Avión
  - A está contenido lógicamente en B
    - Vuelo – PlanificaciónVuelo
  - A es una descripción de B
    - DescripciónDelVuelo – Vuelo
  - A es una línea de una transacción o informe de B
    - TrabajoMantenimiento – RegistroDeMantenimiento
  - A se conoce/registra/recoge/informa/captura en B
    - Reserva – ListadePasajeros
  - A es miembro de B
    - Piloto – CompañíaAérea

Identificación de asociaciones (iii)

- Lista de asociaciones comunes [Larman, 2002] (ii)
  - A es una unidad organizativa de B
    - Mantenimiento – CompañíaAérea
  - A utiliza o gestiona a B
    - Piloto – Avión
  - A se comunica con B
    - AgenteDeReservas – Pasajero
  - A está relacionado con una transacción B
    - Pasajero – Billete
  - A es una transacción relacionada con otra transacción B
    - Reserva – Cancelación
  - A está al lado de B
    - Ciudad – Ciudad
  - A es propiedad de B
    - Avión – CompañíaAérea
  - A es un evento relacionado con B
    - Salida – Vuelo

Identificación de atributos (i)

- Son las propiedades **relevantes** de los objetos individuales
- Antes de identificar los atributos es necesario identificar las asociaciones
  - Relacionar las clases conceptuales con asociaciones no con atributos
- Se pueden utilizar las heurísticas de Abbott (1983) para identificar los atributos
- La mayoría de los atributos simples son los que conocen como tipos de datos primitivos
  - El tipo de un atributo no debería ser un concepto de dominio complejo
  - Los atributos deben ser, generalmente, **tipos de datos**
    - Un tipo de dato para UML implica un conjunto de valores para los cuales no es significativa una identidad única (en el contexto del modelo o sistema en el que se está trabajando) [Rumbaugh et al., 1999]
- Se deberían incluir en un modelo de dominio aquellos atributos para los que los requisitos sugieren o implican una necesidad de registrar la información [Larman, 2002]

Identificación de atributos (ii)

- En caso de duda es mejor definir algo como una clase conceptual en lugar de como un atributo
- Se debe representar lo que podría considerarse, inicialmente, como un tipo de dato como una clase no primitiva si [Larman, 2002]
  - Está compuesta de secciones separadas
  - Habitualmente hay operaciones asociadas con él, como análisis sintáctico o validación
  - Tiene otros atributos
  - Es una cantidad con una unidad
  - Es una abstracción de uno o más tipos con alguna de estas cualidades

Identificación de superclases y subclases (i)

- La definición de una superclase conceptual es más general y abarca más que la definición de una subclase
- Todos los miembros del conjunto de una subclase conceptual son miembros del conjunto de su superclase
- Se tiene que tener la conformidad con la definición de la superclase. Regla del 100% [Larman, 2002]
  - El 100% de la definición de la superclase conceptual se debe poder aplicar a la subclase. La subclase debe ajustarse al 100% de los atributos y asociaciones de la superclase
- Una subclase debe ser miembro del conjunto de la superclase. Regla Es-un [Larman, 2002]
  - Todos los miembros del conjunto de una subclase deben ser miembros del conjunto de su superclase
- Una subclase conceptual correcta debe cumplir [Larman, 2002]
  - La regla del 100% (conformidad en la definición)
  - La regla Es-un (conformidad con la pertenencia al conjunto)

Identificación de superclases y subclases (ii)

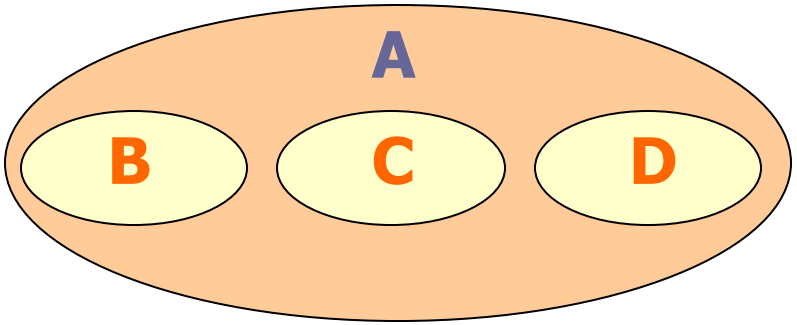
- Razones para especializar una clase conceptual en subclases
  - Una partición de clases conceptuales es una división de las clases conceptuales en subclases disjuntas [Martin y Odell, 1995]
  - Se debería crear una subclase conceptual de una superclase cuando [Larman, 2002]
    - La subclase tiene atributos adicionales de interés
    - La subclase tiene asociaciones adicionales de interés
    - El concepto de la subclase funciona, se maneja, reacciona, o se manipula de manera diferente a la superclase o a otras subclases, de alguna manera que es interesante
    - El concepto de la subclase representa alguna cosa animada que se comporta de manera diferente a la superclase o a otras subclases, de alguna forma que es interesante

Identificación de superclases y subclases (iii)

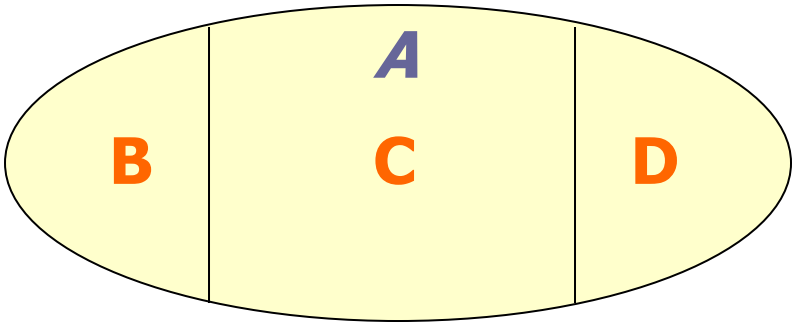
- Razones para definir una superclase conceptual
  - Se aconseja generalizar una superclase común cuando se identifican elementos comunes entre las subclases potenciales
  - Se debería crear una superclase conceptual en una relación de generalización de subclases cuando [Larman, 2002]
    - Las subclases potenciales representen variaciones de un concepto similar
    - Las subclases se ajustan a las reglas del 100% y Es-un
    - Todas las subclases tienen el mismo atributo que se puede factorizar y expresar en la superclase
    - Todas las subclases tienen la misma asociación que se puede factorizar y relacionar con la superclase

Identificación de superclases y subclases (iv)

- Clases conceptuales abstractas
  - Es útil identificar las clases abstractas en el modelo del dominio porque esto restringe las clases que pueden tener instancias concretas
    - Se clarifican las reglas del dominio del problema
  - Si cada miembro de una clase **A** puede ser también miembro de una subclase, entonces **A** es una **clase conceptual abstracta**



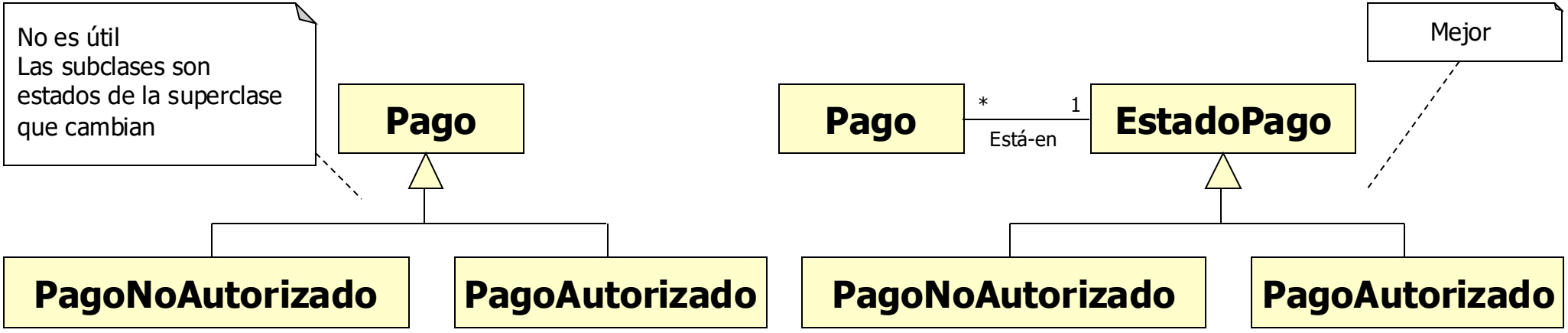
Pueden existir instancias de **A** que no sean instancias de **B**, **C** o **D**. Entonces **A** no es una clase conceptual abstracta



A es una clase conceptual abstracta. Una instancia de **A** debe conformar con una de sus subclases, **B**, **C** o **D**

Identificación de superclases y subclases (v)

- Modelado de los cambios de estado
  - No se debe modelar el estado de un concepto X como subclases de X, sino que se debe seguir una de estas dos estrategias [Larman, 2002]
    - Definir una jerarquía de estados y asociar los estados con X
    - Ignorar la representación de los estados de un concepto en el modelo de dominio; en lugar de esto representar los estados en diagramas de estados



Identificación de relaciones todo-parte (i)

- El patrón todo-parte se da en tres tipos de configuración
  - Ensamblaje y sus partes
    - Suele corresponder a un producto real y sus partes constituyentes
      - Ordenador (placa base, disco...)
  - Contenedor y sus contenidos
    - Variante del anterior. Más relacionado con agrupaciones "lógicas"
      - Oficina (mesas, teléfonos, estanterías...)
  - Grupo y sus miembros
    - Agrupación de intereses
      - Asociación (ACM) y sus miembros (asociados)

Identificación de relaciones todo-parte (ii)

- Una relación todo-parte puede utilizarse cuando [Larman, 2002]
  - El tiempo de vida de la parte está ligado al tiempo de vida del todo
    - Existe una dependencia de creación-eliminación de la parte en el todo
  - Existe un ensamblaje obvio todo-parte físico o lógico
  - Alguna de propiedad del compuesto se propaga a las partes, como la ubicación
  - Las operaciones que se aplican sobre el compuesto se propagan a las partes, como la destrucción, movimiento o grabación
- Dos tipos de relación
  - Agregación
  - Composición

Identificación de relaciones todo-parte (iii)

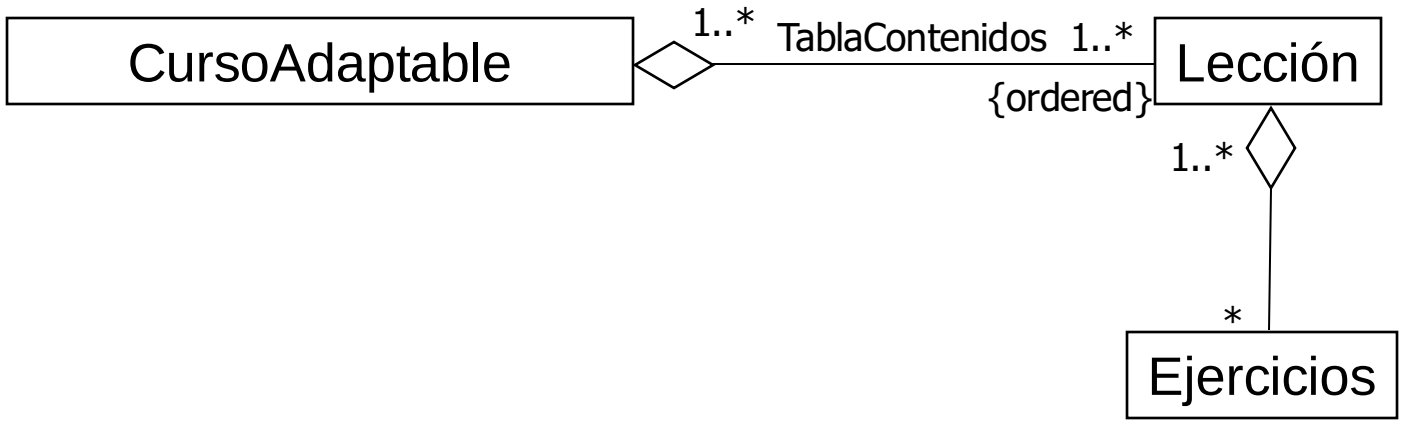
- Agregación vs. Composición (i)
  - Agregación
    - Un objeto agregado es un objeto construido a partir de otros objetos
    - El agregado es mayor que la suma de sus partes
    - Todas las interacciones realizadas con el conjunto de los objetos agregados se realizan a través de la interfaz del objeto agregado
    - Los objetos componentes están ocultos o encapsulados dentro del agregado
  - Composición
    - La composición es una forma fuerte de agregación
    - El ciclo de vida de las partes depende del ciclo de vida del agregado
    - Las partes no existen fuera de su participación en el agregado
    - La pertenencia fuerte implica objetos físicos que se unen para formar el compuesto

Identificación de relaciones todo-parte (iv)

- Agregación vs. Composición (ii)
  - Agregación
    - Multiplicidad en las dos partes de la relación
    - Las partes pueden existir incluso después de que el agregado sea "desmontado" o destruido
    - Las partes pueden cambiar de un agregado a otro
  - Composición
    - La multiplicidad en el "extremo" del compuesto es 1 o 0..1
    - Multiplicidad en el "extremo" de las partes del compuesto
    - Si el agregado se "desmonta" o se destruye las partes no tienen existencia propia
    - Las partes no se pueden mover de una composición a otra

Identificación de relaciones todo-parte (v)

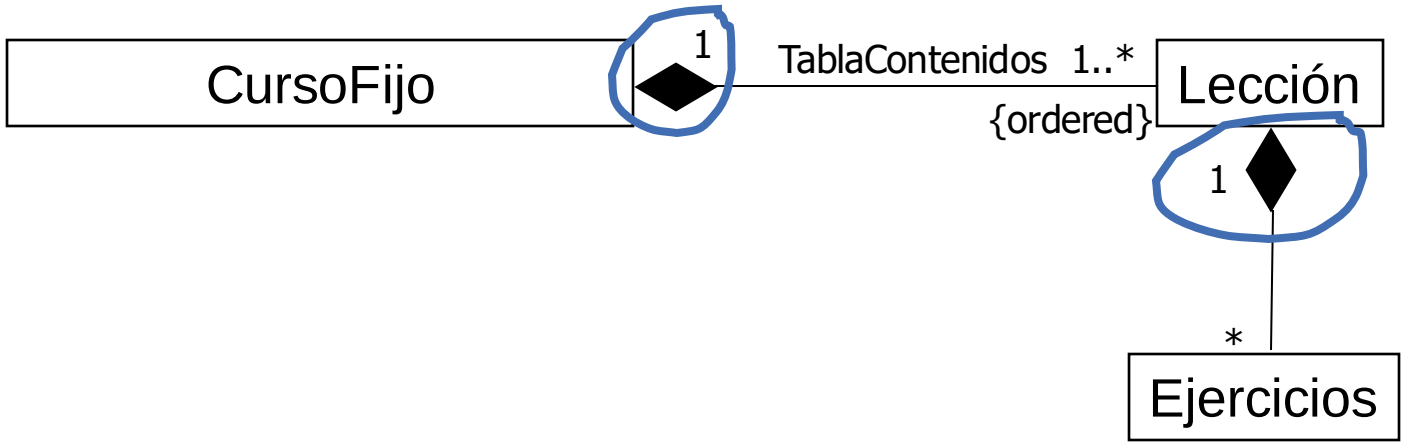
- Ejemplo (i)



  - Cursos adaptables. Se pueden crear combinando lecciones y ejercicios ya existentes creando una tabla de contenidos nueva
  - La tabla de contenidos es única para cada curso
  - La lección aparece en la tabla de contenidos para cada curso que la usa
  - Las lecciones se desarrollan para un curso pero se pueden usar para construir otros cursos
  - Los ejercicios se desarrollan inicialmente para un curso pero pueden ser utilizados con otras lecciones para otros cursos

Identificación de relaciones todo-parte (vi)

- Ejemplo (ii)



- Cursos fijos. Se crean y se entregan. Para crear un nuevo curso todo el material se crea desde de cero
- La tabla de contenidos es única para cada curso
- La tabla de contenidos hace referencia a cada lección del curso. Cada lección solamente aparece en la tabla de contenidos del curso para el que fue desarrollada
- Las lecciones se utilizan únicamente en el curso para el que fueron desarrolladas
- Los ejercicios se utilizan únicamente en las lecciones para las que fueron desarrollados

Identificación de relaciones todo-parte (vii)

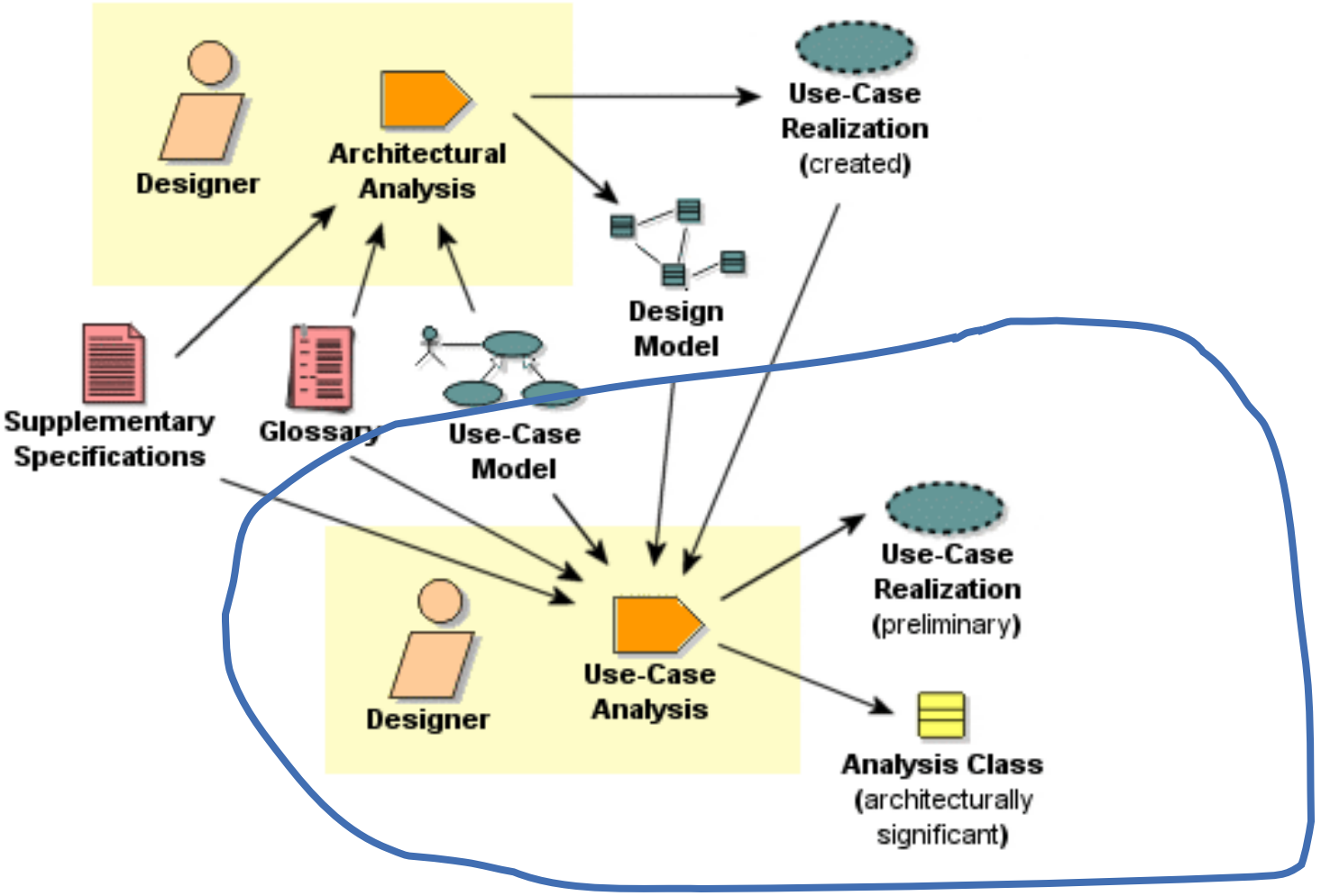
- Identificar y representar las relaciones todo-parte **no** es excesivamente importante en el modelo de dominio
- Descubrir y mostrar relaciones todo-parte si se obtiene alguno de los siguientes beneficios [Larman, 2002]
  - Aclara las restricciones de dominio en cuanto a la existencia que se desea de la parte independiente del todo
    - En la composición, la parte no podría existir fuera del tiempo de vida del todo
  - Ayuda a la identificación de un creador (el compuesto)
  - Las operaciones que se aplican al todo frecuentemente se propagan a las partes

Partición del modelo de dominio

- Para organizar el modelo de dominio en paquetes se deberían poner juntos los elementos que [Larman, 2002]
  - Se encuentran en el mismo área de interés, esto es, estrechamente relacionados por conceptos u objetivos
  - Están juntos en una jerarquía de clases
  - Participan en los mismos casos de uso
  - Están fuertemente asociados

4. Análisis en el Proceso Unificado

Análisis en el Proceso Unificado



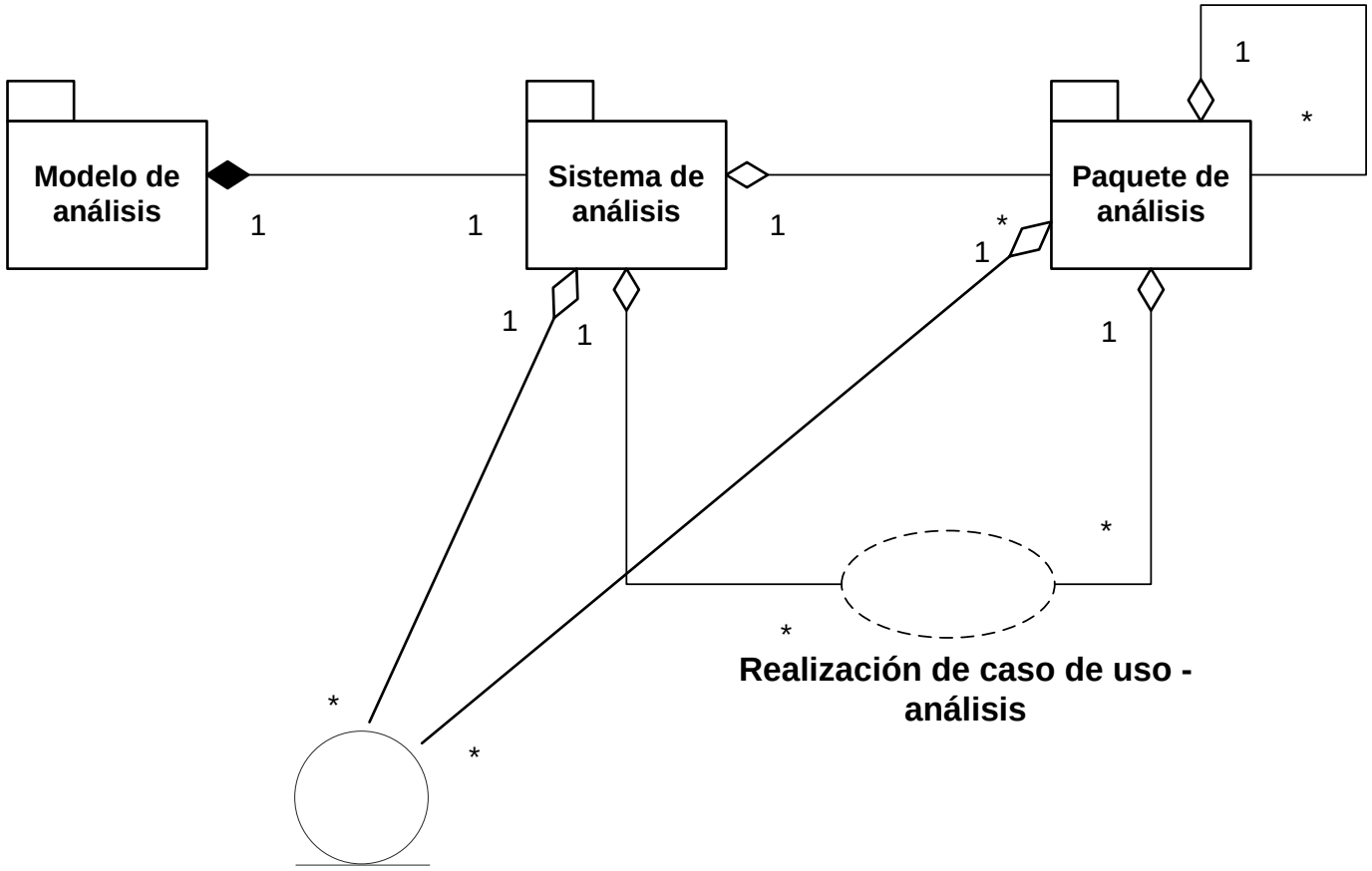
Artefactos propios del análisis en el Proceso Unificado

- Modelo de análisis
- Clase de análisis
- Realización de casos de uso – análisis
- Paquete de análisis
- Descripción de la arquitectura (vista del modelo de análisis)

Modelo de análisis (i)

- Se representa mediante un sistema de análisis que denota el paquete de más alto nivel del modelo
- Se utilizan otros paquetes de análisis para organizar el modelo de análisis en partes más manejables
  - Representan abstracciones de subsistemas y posiblemente capas completas del diseño del sistema
- Las clases de análisis representan abstracciones de clases y posiblemente de subsistemas de diseño del sistema
- Los casos de uso se describen mediante clases de análisis y sus objetos
  - Esto se representa mediante colaboraciones denominadas **realizaciones de caso de uso - análisis**

Modelo de análisis (ii)



[Jacobson et al., 1999]

Clase de análisis (i)

- Representa una abstracción de una o varias clases y/o subsistemas del diseño del sistema
- Características de las clases de análisis
  - Se centran en el tratamiento de los requisitos funcionales
  - Son más evidentes en el contexto del dominio del problema
  - Raramente definen u ofrecen una interfaz en términos de operaciones
  - Pueden definir atributos pero a un nivel bastante alto
  - Participan en relaciones de un claro carácter conceptual
  - Siempre encajan en uno de tres estereotipos básicos
    - **Entidad**
    - **Interfaz**
    - **Control**

Clase de análisis (ii)

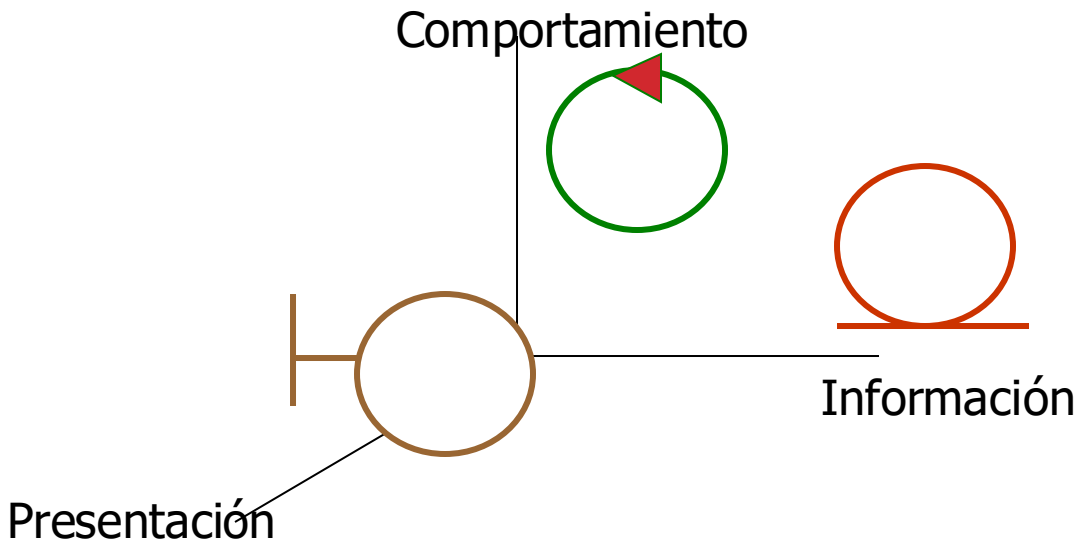
Especificación de objetos en el espacio de información definido por tres ejes

Se eligen tres tipos de objetos a fin de proporcionar una estructura más adaptable

Objetos Entidad: Información persistente sobre la que el sistema realiza un seguimiento

Objetos Interfaz: Representan las interacciones entre el actor y el sistema

Objetos Control: Representan las tareas realizadas por el usuario y soportadas por el sistema

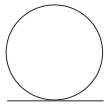


Clase de análisis (iii)

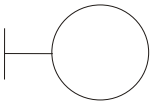
- ¿Por qué estos tres tipos de objetos (entidad, interfaz y control)?
  - Aparecen de forma natural en el texto del caso de uso
  - Se obtienen objetos especializados más pequeños
  - Dan lugar a modelos más resistentes a cambios
    - La interfaz cambia fácilmente
  - Ayudan a la construcción de diagramas de secuencia
  - Los objetos de control sirven de conexión entre los usuarios y los datos almacenados
    - Capturan las reglas de negocio (siempre sujetas a cambios)
    - Sirven de espacio de reserva para garantizar que no se olvida la funcionalidad

Clase de análisis (iii)

- Notación

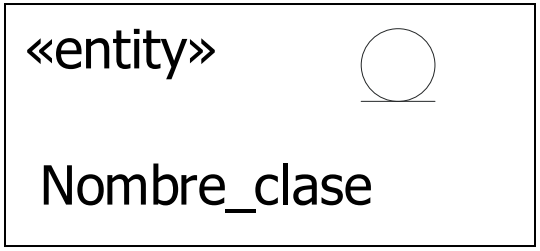


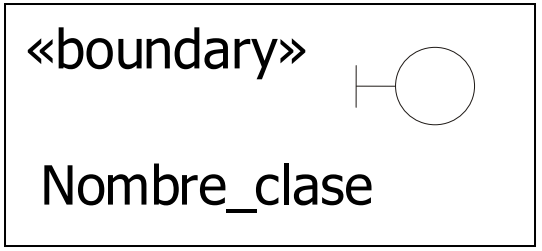
Clase de entidad

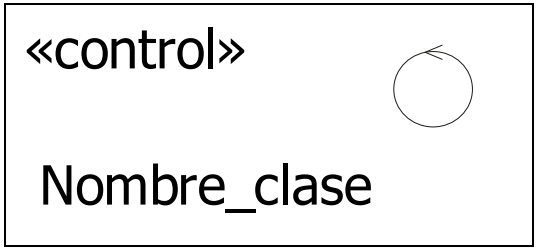


Clase de interfaz

Clase de control







Clase de entidad

- Para modelar la información que el sistema tiene que gestionar se utilizan los objetos entidad
- Esta información permanece en el sistema incluso cuando el caso de uso finaliza
- En el objeto entidad se incorpora, además de la información, el comportamiento asociado a esa información
- Estos objetos se identifican a partir de la descripción de los casos de uso
- Normalmente suelen corresponder a alguno de los conceptos que se manejan en el sistema. **Clases conceptuales**
- Las operaciones identificadas en el objeto tienen que ser suficientes para todas las posibles utilizaciones del objeto
- Hay que evitar en la medida de lo posible que estos objetos sólo sean portadores de información asignando todo el comportamiento dinámico a los objetos control

Clase de interfaz (i)

- Toda la funcionalidad que depende directamente del entorno se asigna a las clases de interfaz
- Modelan las interacciones entre el sistema y sus actores
- Interaccionan con los actores externos al sistema y con las clases del sistema
- Los objetos interfaz se encargan de trasladar las acciones de los actores a eventos en el sistema y éstos en "información" que se presenta al actor
- Representan una abstracción de elementos de la interfaz de usuario (ventanas, formularios, paneles...) o dispositivos (interfaz de impresora, sensores, terminales...)
- Cada actor necesita su/s propia/s interfaz/ces para llevar a cabo sus acciones
- Mantener la descripción a nivel conceptual, es decir, no describir cada botón, ítem de menú... de la interfaz de usuario
- Encapsula y aísla los cambios en la interfaz de usuario

Clase de interfaz (ii)

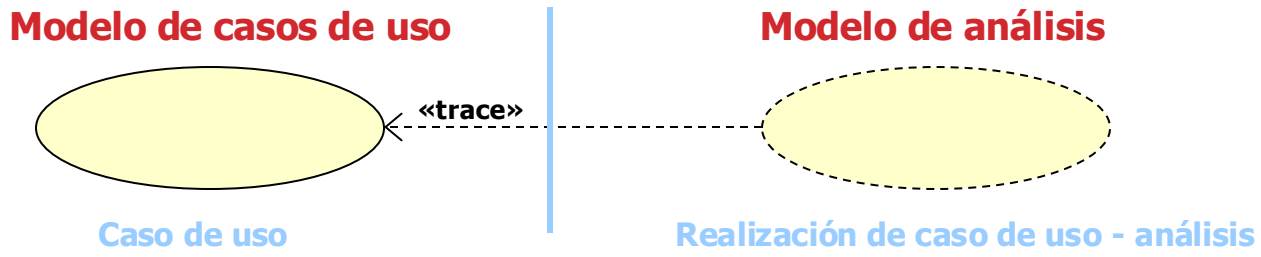
- Estrategias de identificación de clases de interfaz
  - A partir de las descripciones de los casos de uso
  - A partir de las descripciones de las interfaces de usuario
  - A partir de los actores
- Recomendaciones
  - Identificar formularios y ventanas necesarias para la introducción de datos en el sistema
  - Identificar avisos y mensajes a los que tiene que responder el usuario
  - No modelar los aspectos visuales de la interfaz
  - Utilizar siempre términos de usuario para describir las interfaces
- Identificar una **clase de interfaz central** para cada actor humano
  - Representa la "ventana" primaria con la que el actor interacciona
- Identificar una **clase de interfaz central** para cada actor sistema externo
  - Representa la interfaz de comunicación con el sistema externo
- Las clases de interfaz centrales pueden ser **agregaciones** de otras clases de interfaz

Clase de control

- Los objetos de control actúan como unión entre los objetos interfaz y entidad
  - Coordinación entre objetos entidad e interfaz
- No siempre aparecen
  - Normalmente toda la funcionalidad expresada en un caso de uso está asignada a los objetos entidad e interfaz
- Se identifican a partir de los casos de uso
  - Cada caso de uso tiene inicialmente solamente un objeto de control
- El tipo de funcionalidad asignada a estos objetos suele ser el comportamiento relacionado con transacciones o secuencias específicas de control
- El comportamiento de "conexión" entre objetos interfaz y entidad suele asignarse a este tipo de objetos
- Normalmente no se corresponden con entidades reales

Realización de casos de uso – análisis

- Es una colaboración dentro del modelo de análisis que describe cómo se lleva a cabo y se ejecuta un caso de uso en término de las clases de análisis y de sus objetos
- Ofrece una traza directa hacia un caso de uso concreto del modelo de casos de uso
- Una realización de un caso de uso posee
  - Una descripción del flujo de sucesos
  - Diagramas de clase de análisis
  - Diagramas de interacción

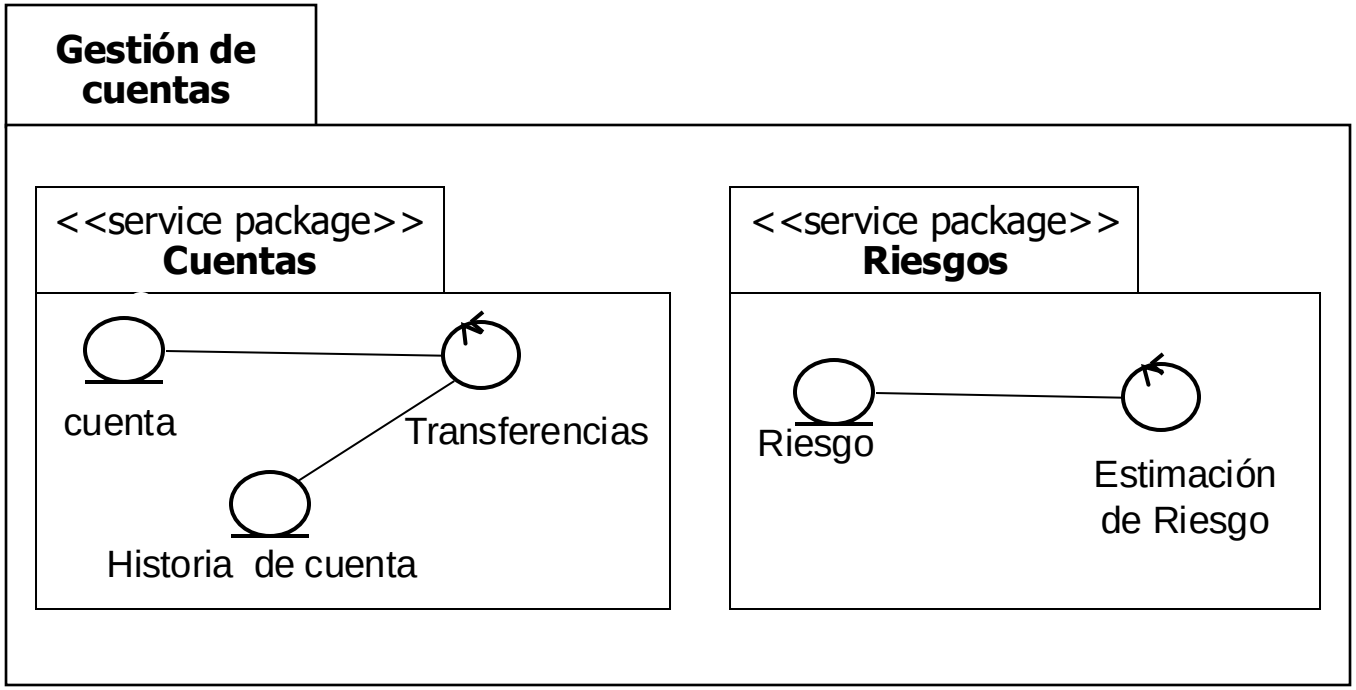


Paquete de análisis (i)

- Es el medio para organizar los artefactos del modelo de análisis en piezas manejables
- Puede constar de clases de análisis, de realización de casos de uso y de otros paquetes del análisis (recursivamente)
- Estos paquetes deben ser cohesivos y débilmente acoplados
- Pueden representar una separación de intereses de análisis
- Deben crearse basándose en los requisitos funcionales en el dominio del problema y deben ser reconocibles por las personas con conocimiento del dominio
- Se suelen convertir en subsistemas en las (dos) capas superiores del modelo de diseño

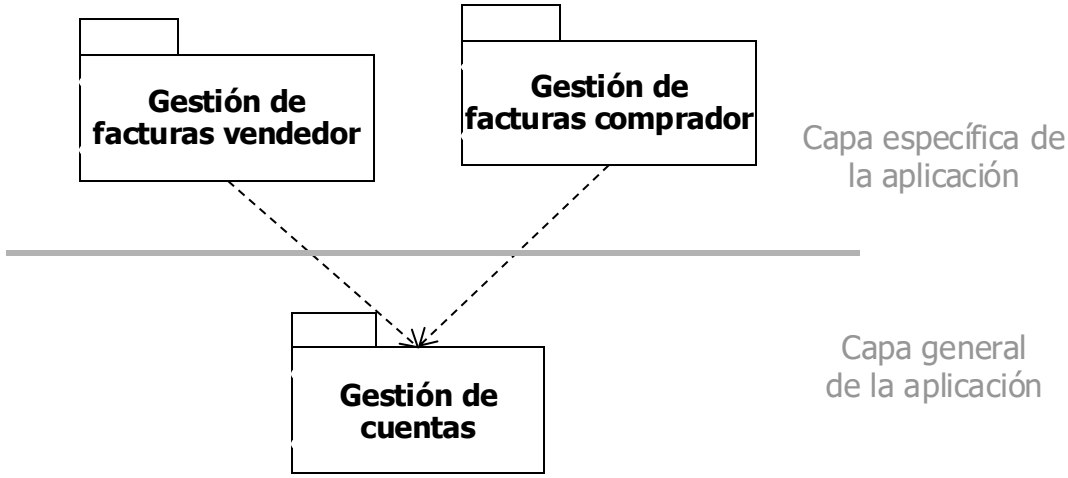
Paquete de análisis (ii)

- Los paquetes de análisis pueden contener paquetes de servicio
- Los paquetes de servicio estructuran el sistema de acuerdo a los servicios que proporciona



Descripción de la arquitectura

- Contiene una vista de la arquitectura del modelo de análisis, que muestra sus artefactos significativos para la arquitectura
  - Descomposición del modelo de análisis en paquetes de análisis y sus dependencias
  - Las clases fundamentales del análisis
  - Realizaciones de casos de uso que describen cierta funcionalidad importante y crítica



Dependencias entre paquetes del análisis

Proceso Unificado: Actividades del análisis

- Completar las descripciones de los casos de uso
  - Incluir los detalles internos de la actividad del sistema en respuesta a las acciones de los actores
- Para cada realización de caso de uso
  - Encontrar las **clases de análisis** a partir del comportamiento del caso de uso
  - Distribuir el comportamiento entre las clases de análisis
- Para cada clase de análisis resultante
  - Describir las responsabilidades
  - Describir atributos y asociaciones
  - Definir atributos
  - Establecer las asociaciones entre las clases de análisis
  - Describir Dependencias entre clases de análisis
- Evaluar el resultado del análisis de casos de uso

https://bit.ly/2JQ8fQ6

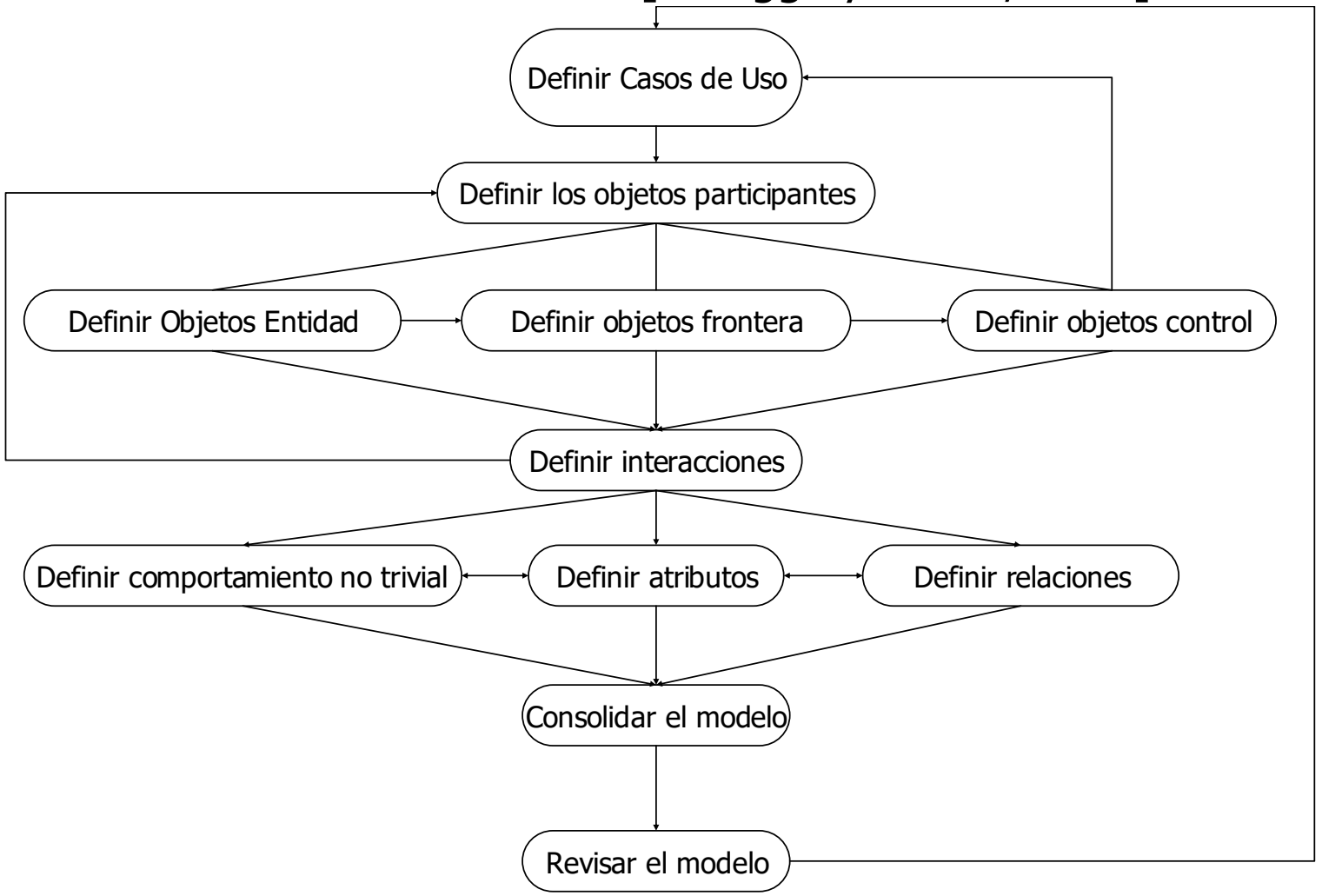
5. Aportaciones principales del tema

Aportaciones principales (i)

- El objetivo del análisis orientado a objetos es la creación de un diagrama de clases conceptual
- Las entradas principales al análisis orientado a objetos son el modelo de casos de uso y el glosario
- La salida del modelo de análisis es un diagrama de clases conceptual
- El análisis casa con el dominio del problema
- Un modelo de dominio muestra las clases conceptuales significativas en un dominio del problema
- Las clases de análisis siempre pueden ser de entidad, interfaz o control

Aportaciones principales (ii)

- Actividades del análisis. Síntesis [Bruegge y Dutoit, 2000]



Aportaciones principales (iii)

Definir Casos de Uso

- Actividades del análisis (i)
  - Detalle de los eventos del sistema de entrada y salida
  - Un **diagrama de secuencia del sistema** (DSS) por caso de uso
  - Diagrama de secuencia creado de forma rápida y fácil
  - El sistema es una "caja negra"
  - ¿Qué hace el sistema? ¿Cómo interactúan los actores externos con el sistema?
  - Un DSS para el escenario principal. Un DSS para los escenarios alternativos complejos o frecuentes
  - Se identifican los límites del sistema
  - Los eventos del sistema (y sus operaciones) deberían expresarse al nivel de intenciones en lugar de en términos del medio de entrada físico o a nivel de elementos de la interfaz de usuario
    - Escanear - NO
    - introducirArtículo - SÍ

Aportaciones principales (iv)

- Actividades del análisis (ii)

Definir los objetos participantes

  - Para cada caso de uso identificar los objetos que participan en cada caso de uso
    - Conjunto posible de elementos del modelo (clases de análisis) capaz de llevar a cabo el comportamiento descrito en el caso de uso
  - Utilizar las técnicas descritas
  - **Utilizar siempre términos del dominio**

Aportaciones principales (v)

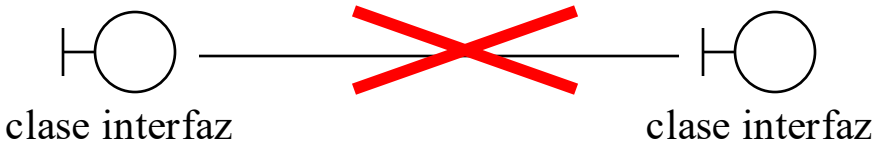
Definir interacciones

- Actividades del análisis (iii)
  - Distribuir el comportamiento entre las clases de análisis
    - Incluir únicamente los objetos necesarios para realizar el caso de uso
    - Se identifican nuevos objetos participantes en el caso de uso
    - Se identifica comportamiento no considerado anteriormente
    - Se centra en el comportamiento de alto nivel
  - Recomendaciones
    - Primera columna el actor que inicia el caso de uso
    - Segunda columna el objeto interfaz con el que interactúa el actor para iniciar el caso de uso
    - Tercera columna el objeto control que gestiona el resto del caso de uso
    - Los objetos de control son creados por objetos interfaz que inician el caso de uso
    - Los objetos interfaz son creados por objetos control
    - Los objetos entidad son accedidos por objetos control

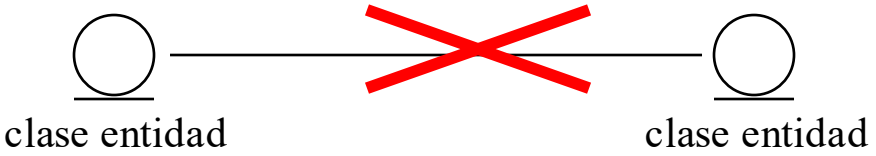
Aportaciones principales (vi)

Definir interacciones

- Actividades del análisis (iv)
  - Heurísticas
    - Actores solamente pueden interaccionar con clases interfaz
    - Clases Interfaz interaccionan, en principio, con clases control y actores



    - Clases Entidad interaccionan, en principio, con clases control



    - Clases Control pueden interaccionar con clases interfaz, entidad y control

Aportaciones principales (vii)

- Actividades del análisis (v)

Definir interacciones

  - Describir responsabilidades
    - Una responsabilidad es cualquier "servicio" que puede ser solicitado a un objeto
    - Una responsabilidad puede implicar una o varias operaciones
    - Una responsabilidad se caracteriza por
      - Las acciones que el objeto puede realizar
      - El conocimiento que el objeto mantiene y proporciona a otros objetos
  - Las responsabilidades se derivan de los mensajes en los diagramas de interacción
    - Pueden aparecer responsabilidades ligadas a requisitos no funcionales
  - Asignar responsabilidades a cada objeto en forma de conjunto de operaciones

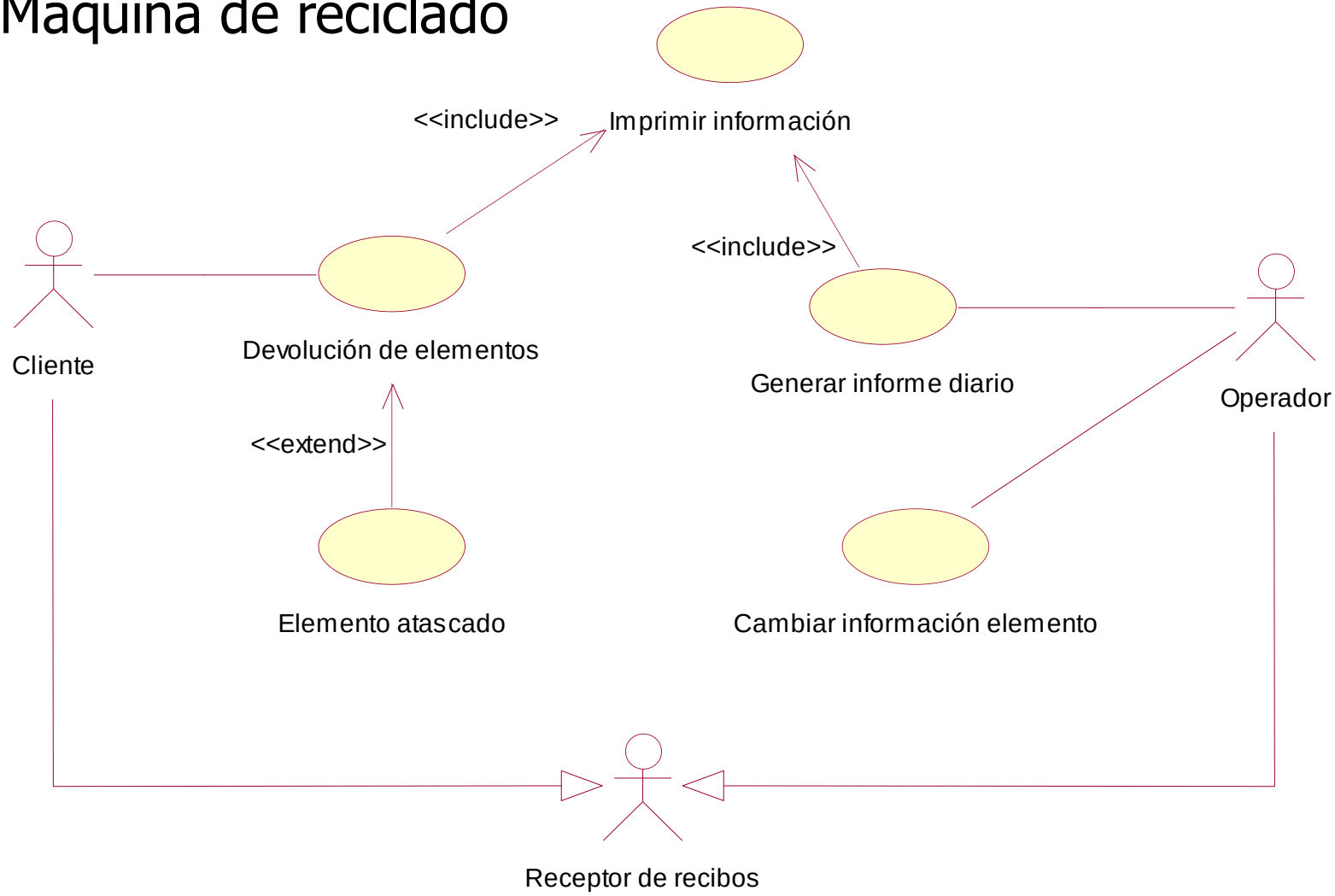
Si una operación aparece en más de un diagrama de secuencia, su comportamiento ha de ser el mismo

https://bit.ly/3pTUm2T

6. Cuestiones y ejercicios

Ejercicio resuelto (i)

- Máquina de reciclado



Ejercicio resuelto (ii)

- Devolución de elementos. Escenario básico
  - Inicio: El cliente desea devolver latas, botellas y envases
  - El caso de uso comienza cuando el cliente presiona el "botón inicio" en el panel del cliente. Los sensores incorporados en el panel se activan
  - El cliente puede devolver elementos (bote, botella, envase) a través del panel de cliente
  - Los sensores informan al sistema que un objeto ha sido insertado, calibran el elemento depositado y devuelven el resultado al sistema
  - El sistema utiliza el resultado medido para determinar el tipo de elemento devuelto
  - El total diario de elementos depositados se incrementa, así como el total de elementos que el cliente ha depositado
  - Cuando el cliente ha depositado todos los elementos a devolver solicita el recibo presionando el "botón recibo"
  - El sistema genera un recibo con la información recogida por cada tipo de elemento depositado
  - Se imprime el recibo y el caso de uso finaliza

Ejercicio resuelto (iii)

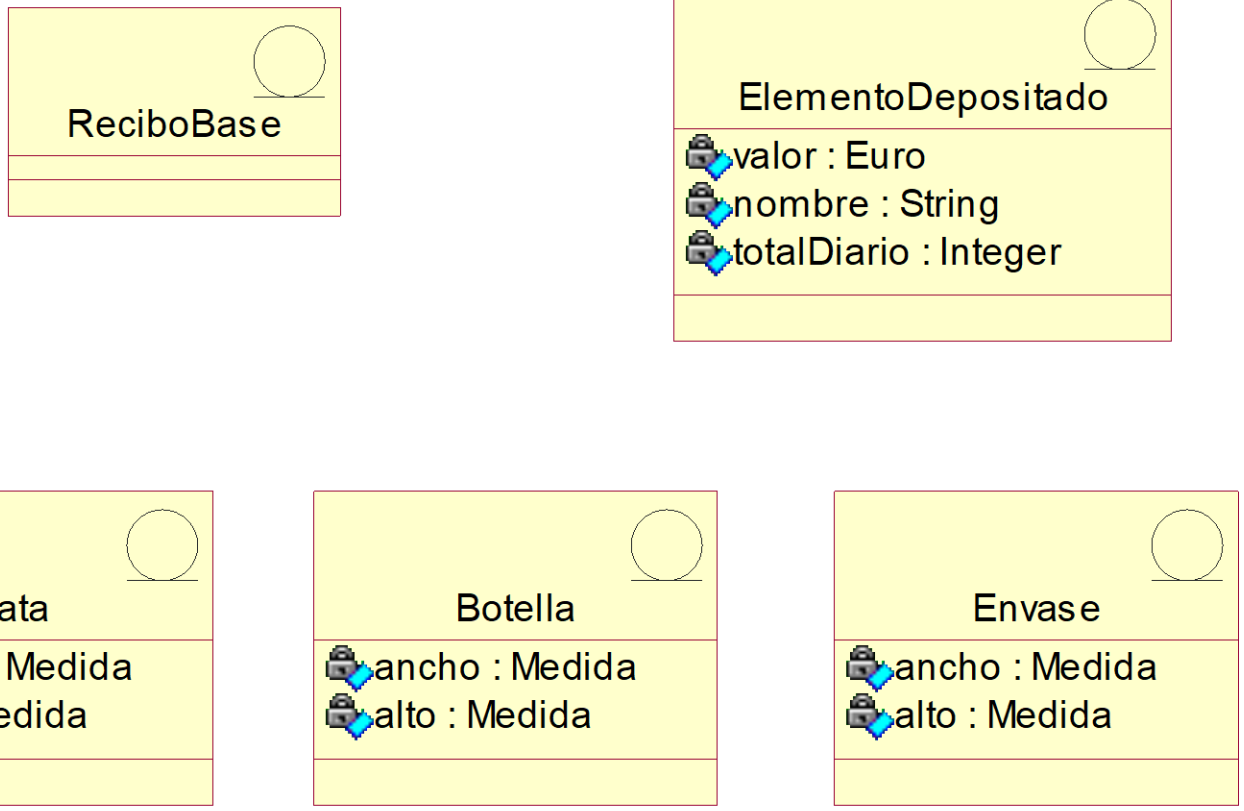
- Devolución de elementos. Objetos de dominio
  - Inicio: El **cliente** desea devolver latas, botellas y envases
  - El caso de uso comienza cuando el cliente presiona el "botón inicio" en el panel del cliente. Los sensores incorporados en el panel se activan
  - El cliente puede devolver **elementos** (**bote**, **botella**, **envase**) a través del panel de cliente
  - Los sensores informan al sistema que un objeto ha sido insertado, calibran el elemento depositado y devuelven el resultado al sistema
  - El sistema utiliza el resultado medido para determinar el tipo de elemento devuelto
  - El total diario de elementos depositados se incrementa, así como el total de elementos que el cliente ha depositado
  - Cuando el cliente ha depositado todos los elementos a devolver solicita el recibo presionando el "botón recibo"
  - El sistema genera un recibo con la información recogida por cada tipo de elemento depositado
  - Se imprime el **recibo** y el caso de uso finaliza

Ejercicio resuelto (iv)

- Devolución de elementos. Clases de entidad (i)
  - Inicio: El **cliente** desea devolver latas, botellas y envases
  - El caso de uso comienza cuando el cliente presiona el "botón inicio" en el panel del cliente. Los sensores incorporados en el panel se activan
  - El cliente puede devolver **elementos** (**bote**, **botella**, **envase**) a través del panel de cliente
  - Los sensores informan al sistema que un objeto ha sido insertado, calibran el elemento depositado y devuelven el resultado al sistema
  - El sistema utiliza el *resultado medido para determinar el tipo de elemento devuelto*
  - El *total diario de elementos depositados se incrementa*, así como el *total de elementos que el cliente ha depositado*
  - Cuando el cliente ha depositado todos los elementos a devolver solicita el recibo presionando el "botón recibo"
  - El sistema compila la información que ha de imprimirse en el recibo. Por cada tipo de elemento depositado extrae *su valor de devolución y el número de elementos depositados* por el cliente actual
  - Se imprime el **recibo** con el detalle y el total de los elementos devueltos y el caso de uso finaliza

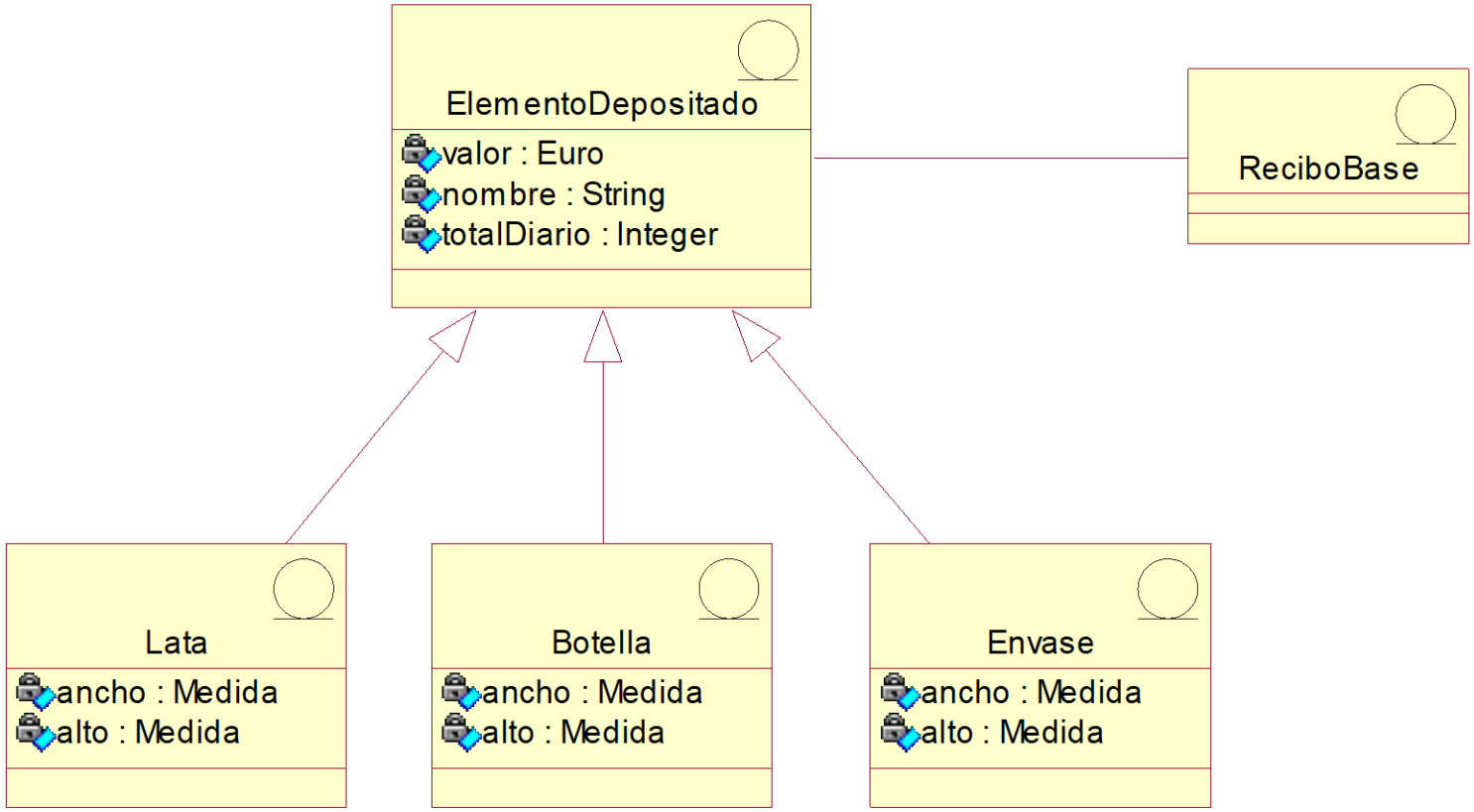
Ejercicio resuelto (v)

- Devolución de elementos. Clases de entidad (ii)



Ejercicio resuelto (vi)

- Devolución de elementos. Clases de entidad (iii)



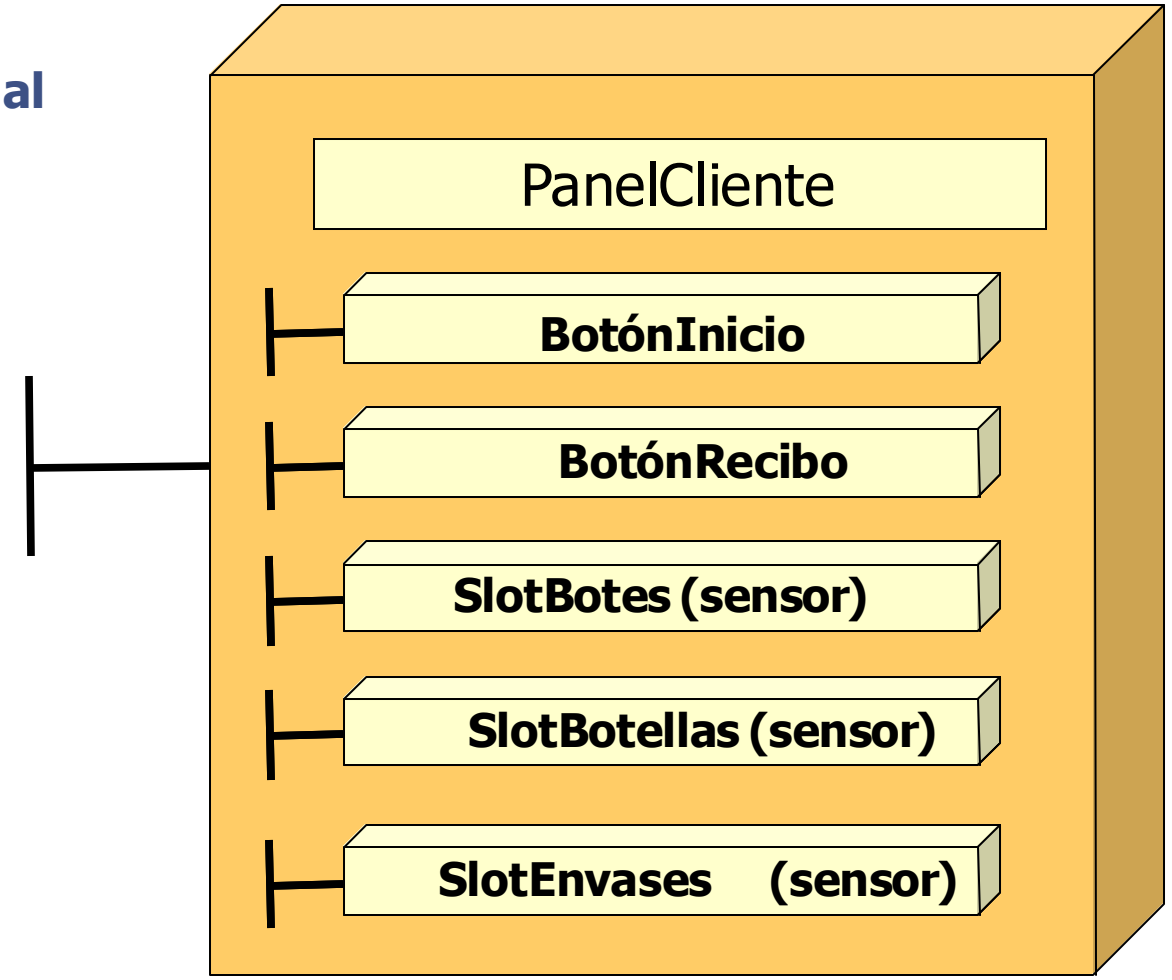
Ejercicio resuelto (vii)

- Devolución de elementos. Clases de interfaz (i)
  - Inicio: El cliente desea devolver latas, botellas y envases
  - El caso de uso comienza cuando el cliente presiona el "**botón inicio**" en el **panel del cliente**. Los **sensores** incorporados en el panel se activan
  - El cliente puede devolver elementos (**bote**, **botella**, **envase**) a través del panel de cliente
  - Los sensores informan al sistema que un objeto ha sido insertado, calibran el elemento depositado y devuelven el resultado al sistema
  - El sistema utiliza el resultado medido para determinar el tipo de elemento devuelto
  - El total diario de elementos depositados se incrementa, así como el total de elementos que el cliente ha depositado
  - Cuando el cliente ha depositado todos los elementos a devolver solicita el recibo presionando el "**botón recibo**"
  - El sistema genera un recibo con la información recogida por cada tipo de elemento depositado
  - Se imprime el recibo y el caso de uso finaliza

Ejercicio resuelto (viii)

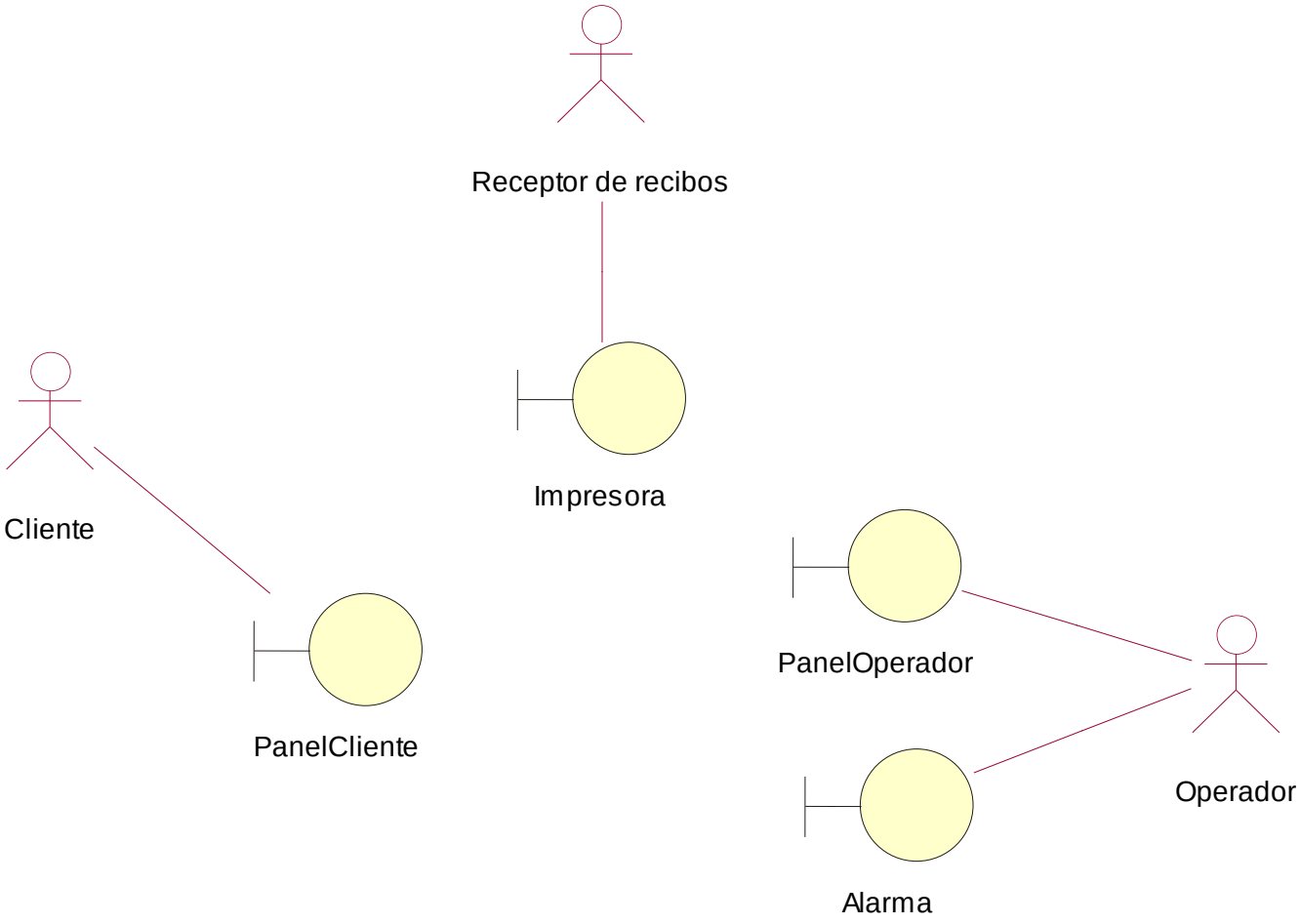
- Devolución de elementos. Clases de interfaz (ii)

Clase interfaz principal



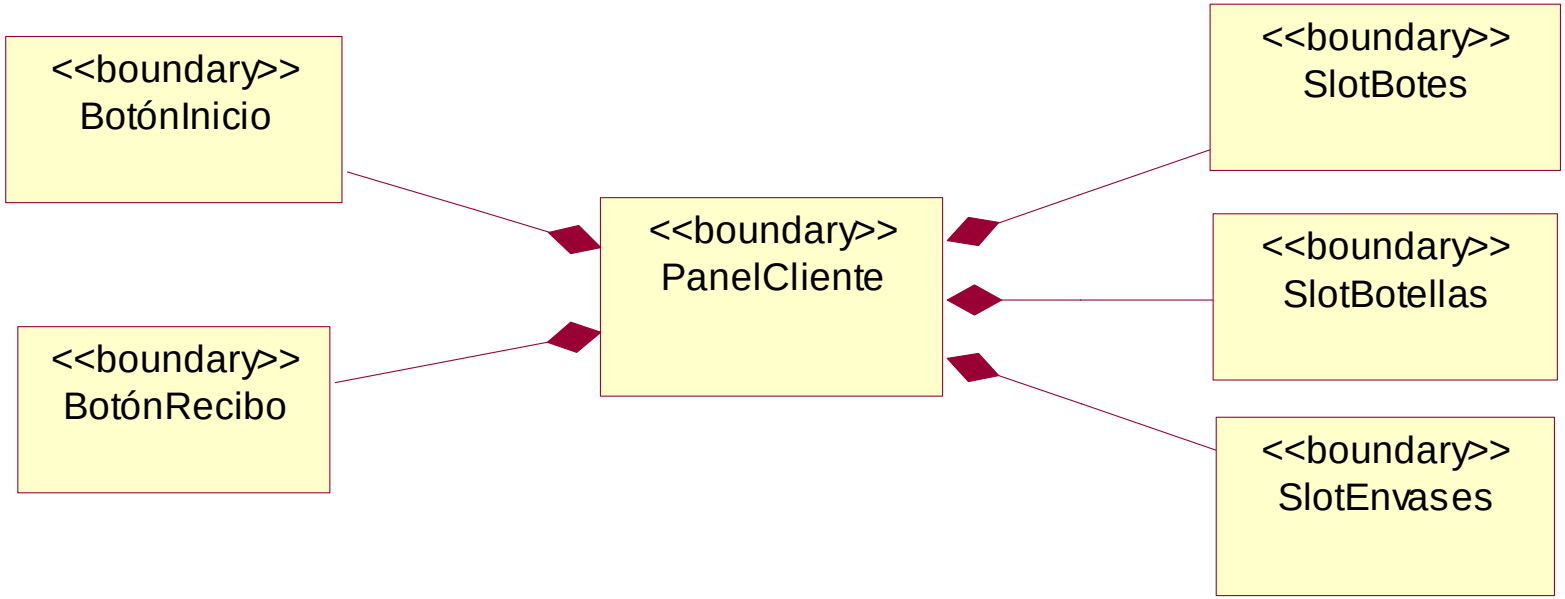
Ejercicio resuelto (ix)

- Devolución de elementos. Clases de interfaz (iii)



Ejercicio resuelto (x)

- Devolución de elementos. Clases de interfaz (iv)

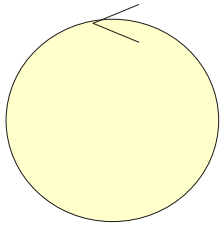


Ejercicio resuelto (xi)

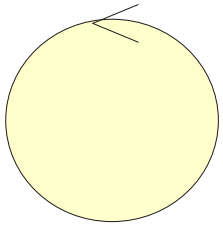
- Devolución de elementos. Clases de control (i)
  - Inicio: El cliente desea devolver latas, botellas y envases
  - El caso de uso comienza cuando el cliente presiona el "botón inicio" en el panel del cliente. Los sensores incorporados en el panel se activan
  - El cliente puede devolver elementos (bote, botella, envase) a través del panel de cliente
  - Los sensores informan al sistema que un objeto ha sido insertado, *calibran el elemento depositado* y *devuelven el resultado al sistema*
  - El sistema utiliza el *resultado medido para determinar* el tipo de elemento devuelto
  - El *total diario de elementos depositados* se incrementa, así como el total de elementos que el cliente ha depositado
  - Cuando el cliente ha depositado todos los elementos a devolver solicita el recibo presionando el "botón recibo"
  - El *sistema compila la información que ha de imprimirse en el recibo*. Por cada tipo de elemento depositado extrae su valor de devolución y el número de elementos depositados por el cliente actual
  - *Se imprime el recibo con el detalle y el total de los elementos devueltos* y el caso de uso finaliza

Ejercicio resuelto (xii)

- Devolución de elementos. Clases de control (ii)



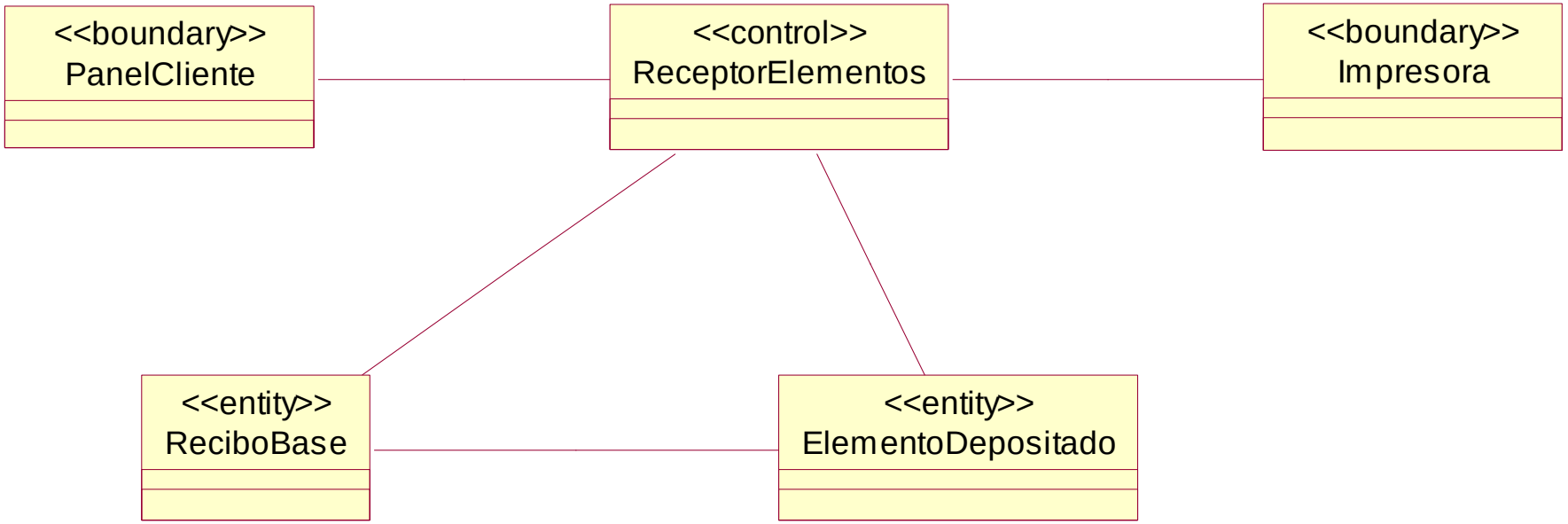
ReceptorElementos



GeneradorInformes

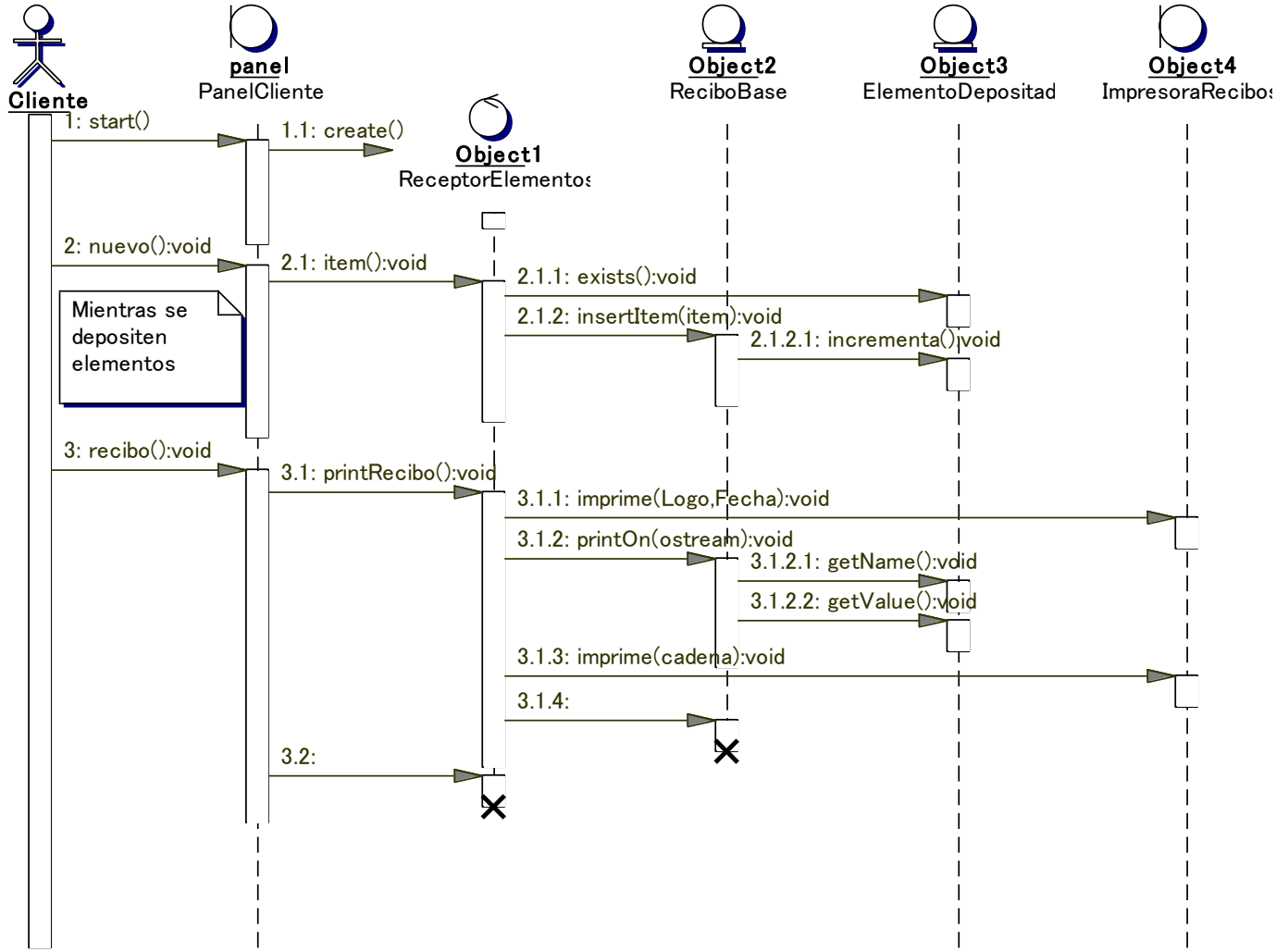
Ejercicio resuelto (xiii)

- Devolución de elementos. Clases de control (iii)



Ejercicio resuelto (xiv)

- Devolución de elementos. Diagrama de Secuencia



Cuestiones y ejercicios

- Realizar el modelo de dominio de una biblioteca
- Realizar el modelo de dominio para una empresa de transporte
- Realizar el modelo de análisis para un vídeo club
- Realizar el modelo de análisis para un sistema de ventas

7. Lecturas complementarias

Lecturas complementarias

- B. Bruegge y A. H. Dutoit, *Object-oriented software engineering. Using UML, patterns, and Java*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2010
  - Es interesante el capítulo 5 "*Analysis*"
- L. A. Maciaszek, *Requirements analysis and system design: Developing information systems with UML*. Essex, UK: Addison-Wesley Longman Ltd., 2001
  - Cabe destacar el capítulo 4 "*Requirements Specification*" y el capítulo 5 "*Advanced Analysis*"
- Ministerio de las Administraciones Públicas, *Métrica v3*, Madrid, España: Ministerio de las Administraciones Públicas, 2001. [Online]. Disponible en: https://d66z.short.gy/MEmYhr
  - Es interesante destacar la parte de análisis orientado a objetos de esta metodología
- J. J. Odell, *Advanced object-oriented analysis and design using UML* (SIGS Reference Library). SIGS Books & Multimedia, 1998
  - Colección de artículos relacionados con el modelado de objetos
- J. Rumbaugh, *OMT insights. Perspectives on Mmodeling from the Journal of Object-Oriented Programming*. New York, NY, USA: SIGS Books Publications, 1996
  - Colección de artículos relacionados con el modelado de objetos
- J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy y W. Lorensen, *Modelado y diseño orientados a objetos. Metodología OMT*. Madrid, España: Prentice-Hall, 1996
  - De este libro clásico cabe destacar los capítulos 1, 3, 4, 7 y 8

https://unsplash.com/photos/7DMkvNbIkpw

8. Referencias

Referencias

[Abbott, 1983] Abbott, R. J. "*Program Design by Informal English Descriptions*". Communications of the ACM, 26(11):882-894. November 1983

[Bruegge y Dutoit, 2000] Bruegge, B., Dutoit, A. "*Object-Oriented Software Engineering*". Prentice-Hall, 2000

[Coad y Yourdon, 1990] Coad, P., Yourdon, E. "*OOA - Object-Oriented Analysis*". Prentice-Hall, 1990

[Fowler, 1996] Fowler, M. "*Analysis Patterns: Reusable Object Models*". Addison-Wesley, 1996

[Hay, 1996] Hay, D. "*Data Model Patterns: Conventions of Thought*". Dorset House, 1996

[IEEE, 1999] IEEE. "*IEEE Software Engineering Standards Collection 1999 Edition. Volume 1: Customer and Terminology Standards*". IEEE Computer Society Press, 1999

[Jacobson et al., 1999] Jacobson, I., Booch, G., Rumbaugh, J. "*The Unified Software Development Process*". Object Technology Series. Addison-Wesley, 1999

[Larman, 2002] Larman, C. "*Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process*". 2nd Ed. Prentice Hall, 2002

[Martin y Odell, 1995] Martin, J., Odell, J. "*Object-Oriented Methods: A Foundation*". Prentice-Hall, 1995

[Monarchi y Puhr, 1992] Monarchi, D. E., Puhr, G. I. "*A Research Typology for Object-Oriented Analysis and Design*". Communications of the ACM, 35(9):35-47. September 1992

[RAE, 2024] Real Academia Española "*Diccionario de la Lengua Española*". Versión electrónica 23.8. http://www.rae.es. [Última vez visitado, 8-2-2025]. 2024

[Rumbaugh et al., 1999] Rumbaugh, J., Jacobson, I., Booch, G. "*The Unified Modeling Language. Reference Manual*". Addison-Wesley, Object Technology Series, 1999

[Shlaer y Mellor, 1988] Shlaer, S., Mellor S. J. "*Object-Oriented Analysis: Modeling the World in Data*". Yourdon Press, 1988

INGENIERÍA DE SOFTWARE I

Tema 7 - Análisis Orientado a Objetos

Grado en Ingeniería Informática
Fecha de última modificación: 9-2-2025

Dr. Francisco José García-Peñalvo / fgarcia@usal.es
Dra. Alicia García-Holgado / aliciagh@usal.es
Dra. Andrea Vázquez-Ingelmo / andreavazquez@usal.es

Departamento de Informática y Automática
Universidad de Salamanca