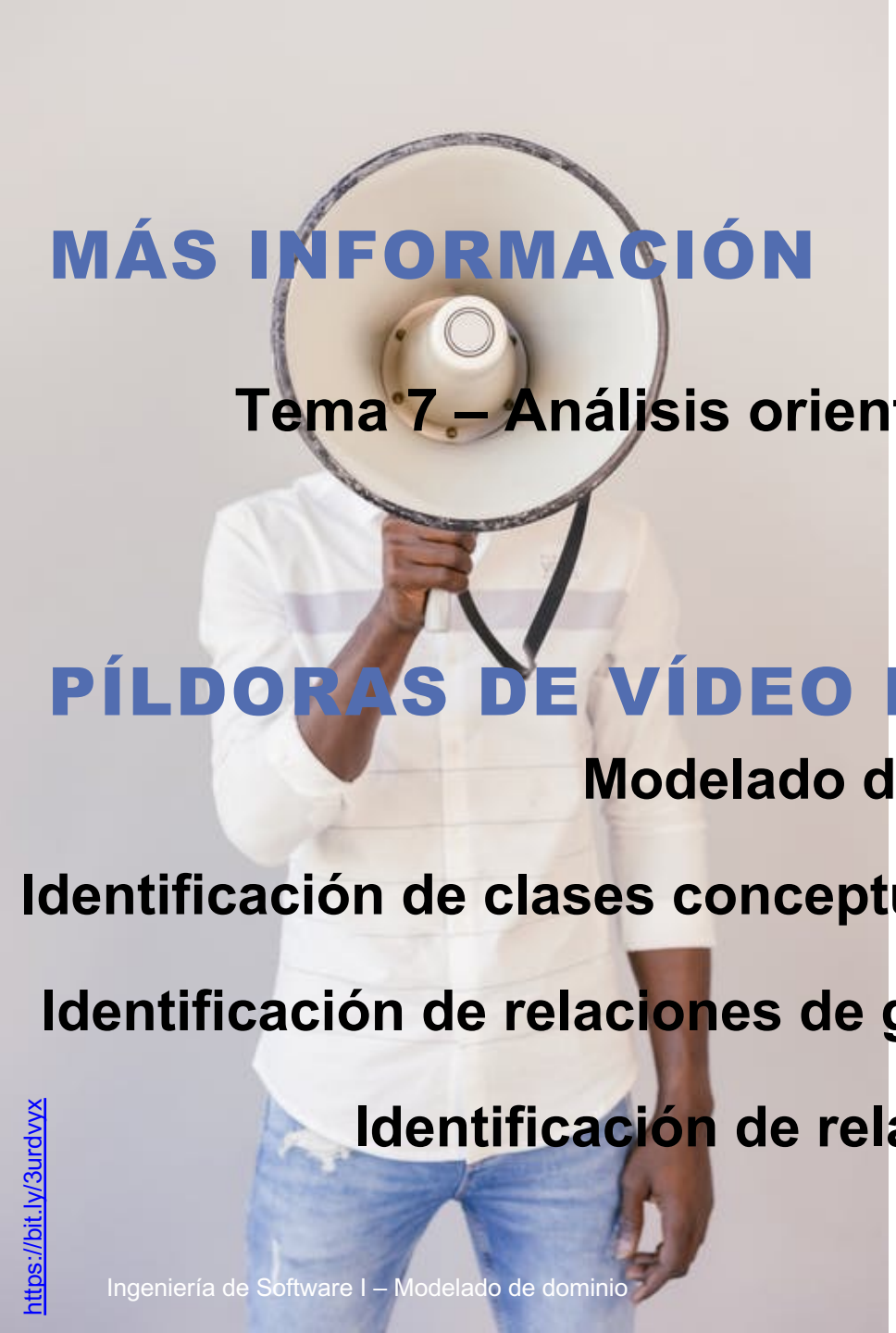
MODELO DE DOMINIO

INGENIERÍA DE SOFTWARE I

2º DE GRADO EN INGENIERÍA INFORMÁTICA
CURSO 2023/2024

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

Departamento de Informática y Automática
Universidad de Salamanca



MÁS INFORMACIÓN

Tema 7 – Análisis orientado a objetos

(García-Peñalvo et al., 2024)

PÍLDORAS DE VÍDEO RELACIONADAS

Modelado de dominio

(García-Peñalvo et al., 2021d)

Identificación de clases conceptuales, asociaciones y atributos

(García-Peñalvo et al., 2021a)

Identificación de relaciones de generalización/especialización

(García-Peñalvo et al., 2021b)

Identificación de relaciones todo-parte

(García-Peñalvo et al., 2021c)

https://bit.ly/3urdvyx

ÍNDICE

- Concepto
- Guías para hacer un modelo de dominio
- Identificación de clases conceptuales
- Identificación de asociaciones
- Identificación de atributos
- Identificación de relaciones de generalización
- Identificación de relaciones todo-parte

CONCEPTO

- 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

CONCEPTO

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, 2004)

GUÍAS PARA HACER UN MODELO DE DOMINIO

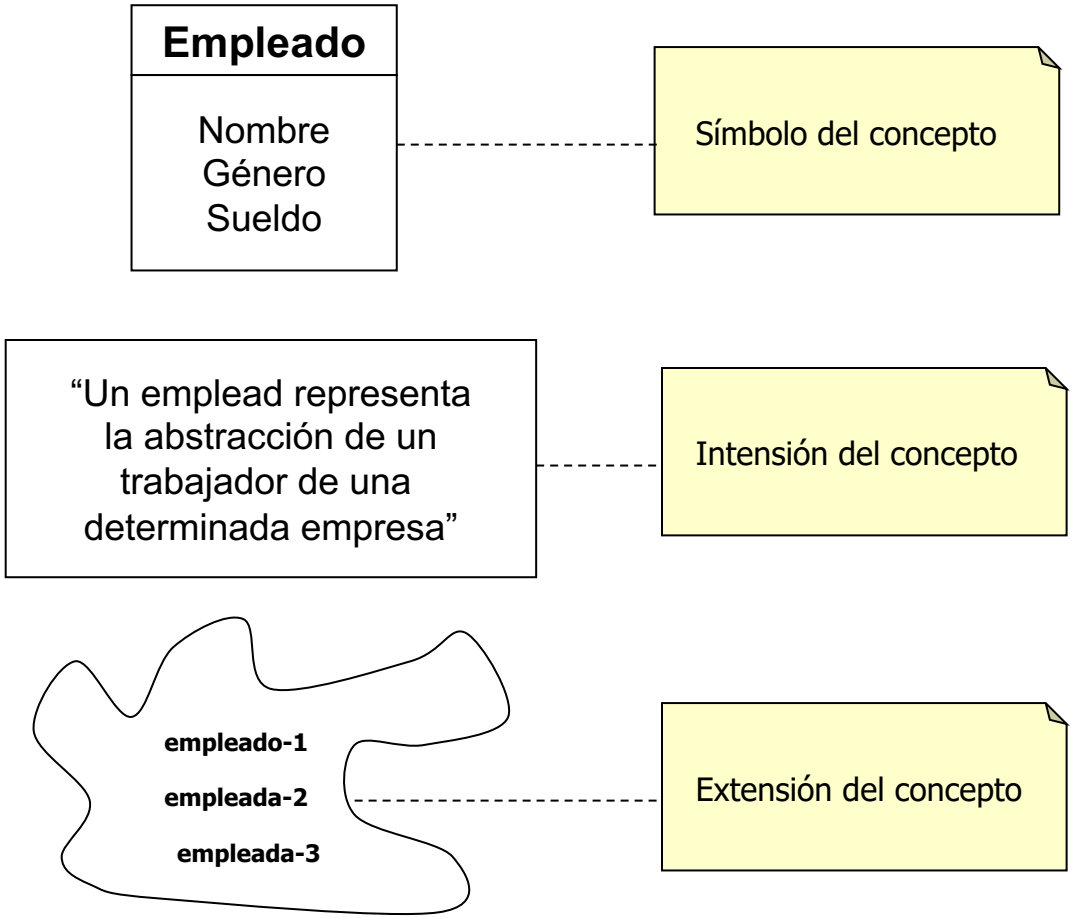
- Listar las clases conceptuales candidatas relacionadas con los requisitos actuales en estudio
- Representar las clases en un modelo de dominio
- Añadir las asociaciones necesarias para registrar las relaciones que hay que mantener en memoria
- Añadir los atributos necesarios para satisfacer los requisitos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

(Larman, 2004)

IDENTIFICACIÓN DE CLASES CONCEPTUALES

- 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, 1997)
  - **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



IDENTIFICACIÓN DE CLASES CONCEPTUALES

- 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, 2004)
  - 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, 1997; Hay, 1996)
    - Un patrón de análisis es un modelo de dominio parcial y existente que ha sido creado por expertos

IDENTIFICACIÓN DE CLASES CONCEPTUALES

Método: Lista de categorías (Shlaer y Mellor, 1988; Larman, 2004)

- 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

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

Método (Coad y Yourdon, 1990). De las candidatas se incluyen en el modelo aquellas clases 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 ASOCIACIONES

Se deben de incluir las siguientes asociaciones en un modelo del dominio (Larman, 2004)

- 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, 2004)

- 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

Lista de asociaciones comunes (Larman, 2004)

- 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

Lista de asociaciones comunes (Larman, 2004)

- 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

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

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., 2005)

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, 2004)

IDENTIFICACIÓN DE ATRIBUTOS

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, 2004)

- 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 RELACIONES DE GENERALIZACIÓN

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 debe tener la conformidad con la definición de la superclase. Regla del 100% (Larman, 2004)

- **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, 2004)

- **Todos los miembros del conjunto de una subclase deben ser miembros del conjunto de su superclase**

Una subclase conceptual correcta debe cumplir (Larman, 2004)

- La regla del 100% (conformidad en la definición)
- La regla Es-un (conformidad con la pertenencia al conjunto)

IDENTIFICACIÓN DE RELACIONES DE GENERALIZACIÓN

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, 1997)
- Se debería crear una subclase conceptual de una superclase cuando (Larman, 2004)
  - 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 RELACIONES DE GENERALIZACIÓN

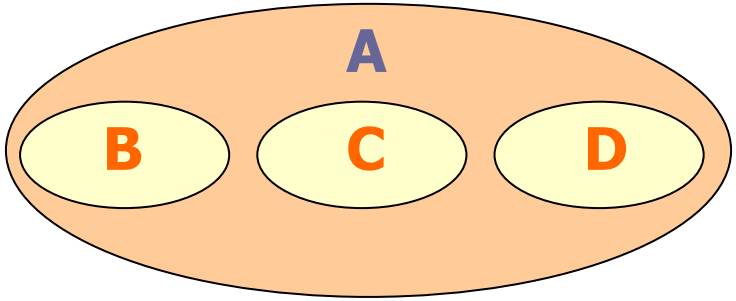
Razones para definir una superclase conceptual

- 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, 2004)
  - 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

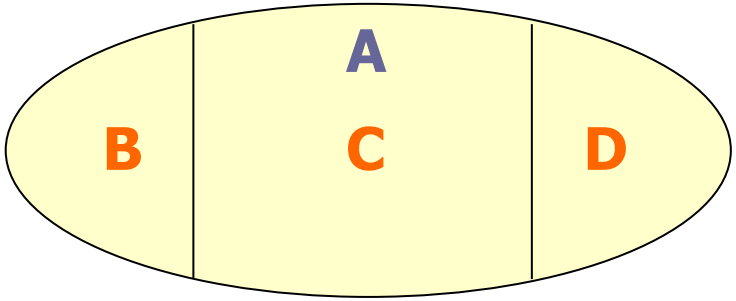
IDENTIFICACIÓN DE RELACIONES DE GENERALIZACIÓN

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**

UML PERMITE DIFERENTES RESTRICCIONES

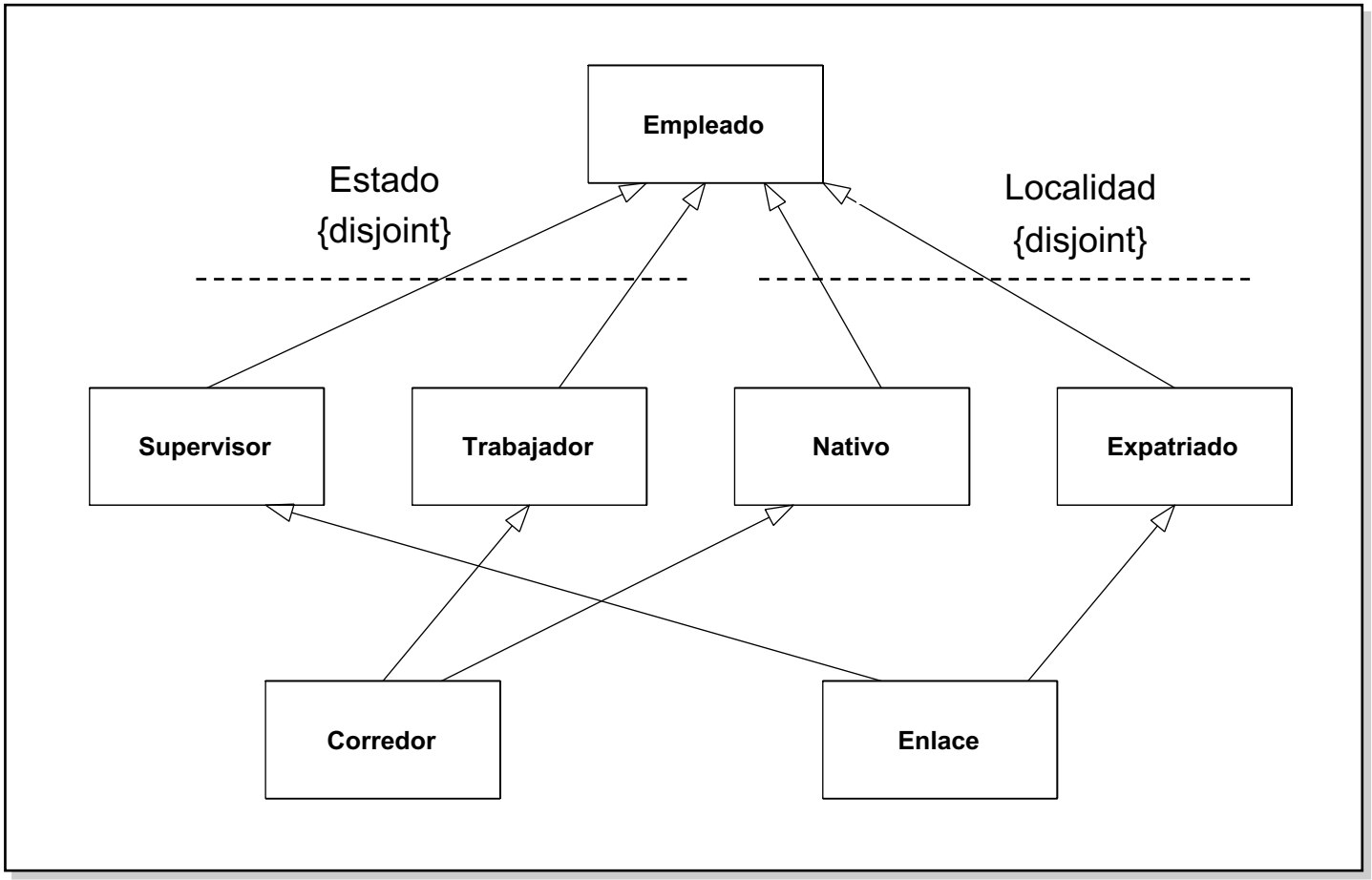
Restricciones de los conjuntos de generalización

- ***Disjoint*** (disjunto) – Ningún elemento puede tener dos hijos en el conjunto como antecesores (en una situación de generalización múltiple). Ninguna instancia puede ser una instancia directa o indirecta de dos de los hijos (en una semántica múltiple de la clasificación)
- ***Overlapping*** (solapado) – Un elemento puede tener dos o más hijos en el conjunto de antecesores. Una instancia puede ser una instancia de dos o más hijos
- ***Complete*** (completo) – Todos los hijos posibles se han enumerado en el conjunto y no puede ser agregado ninguna más
- ***Incomplete*** (incompleto) – No se ha enumerado todavía todos los hijos posibles en el conjunto. Se esperan más hijos o se conocen pero no se han declarado aún

Por defecto: **{incomplete, disjoint}**

Ideal: **{complete, disjoint}**

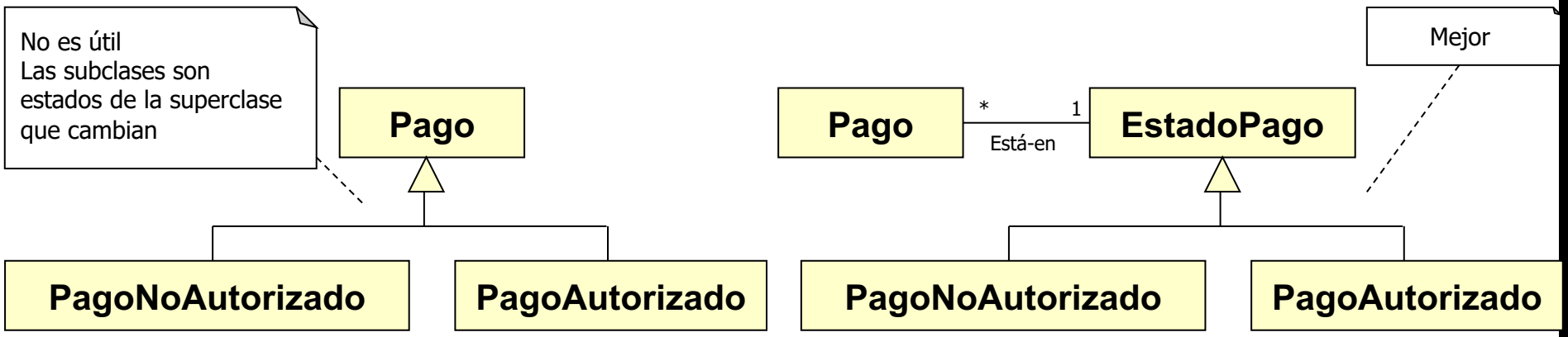
NO TODO LO QUE PERMITE HACER UML IMPLICA UNA BUENA PRÁCTICA DE MODELADO CONCEPTUAL



IDENTIFICACIÓN DE RELACIONES DE GENERALIZACIÓN

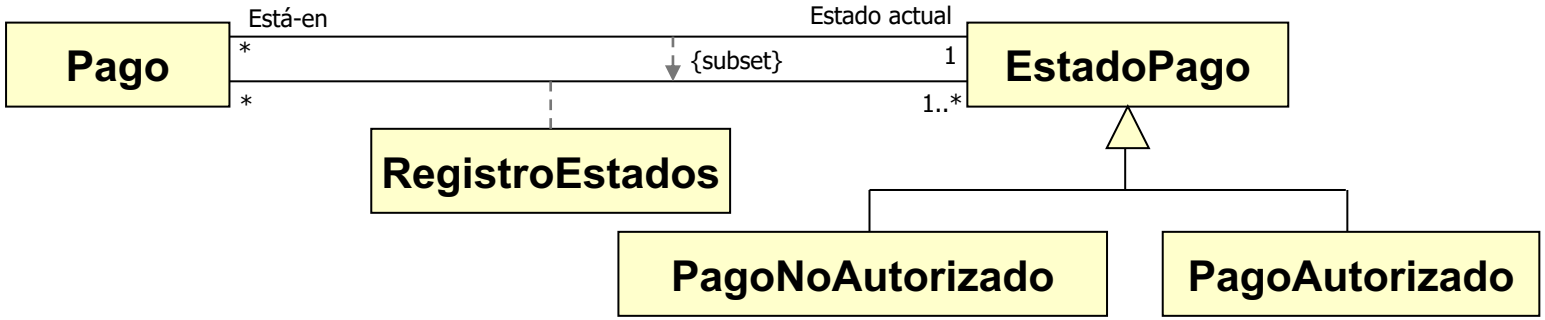
Modelado de los cambios de estado

- 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, 2004)
  - 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



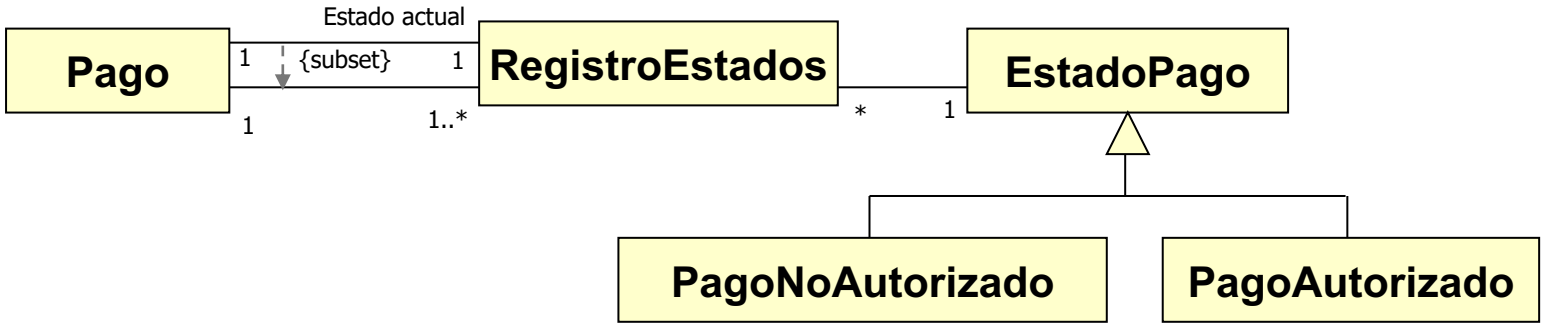
IDENTIFICACIÓN DE RELACIONES DE GENERALIZACIÓN

Modelo con histórico de estados (pero en el que se sabe que nunca se pasará dos veces por el mismo estado)



IDENTIFICACIÓN DE RELACIONES DE GENERALIZACIÓN

Modelo con histórico de estados (pero en el que no se puede garantizar que nunca se pasará dos veces por el mismo estado)



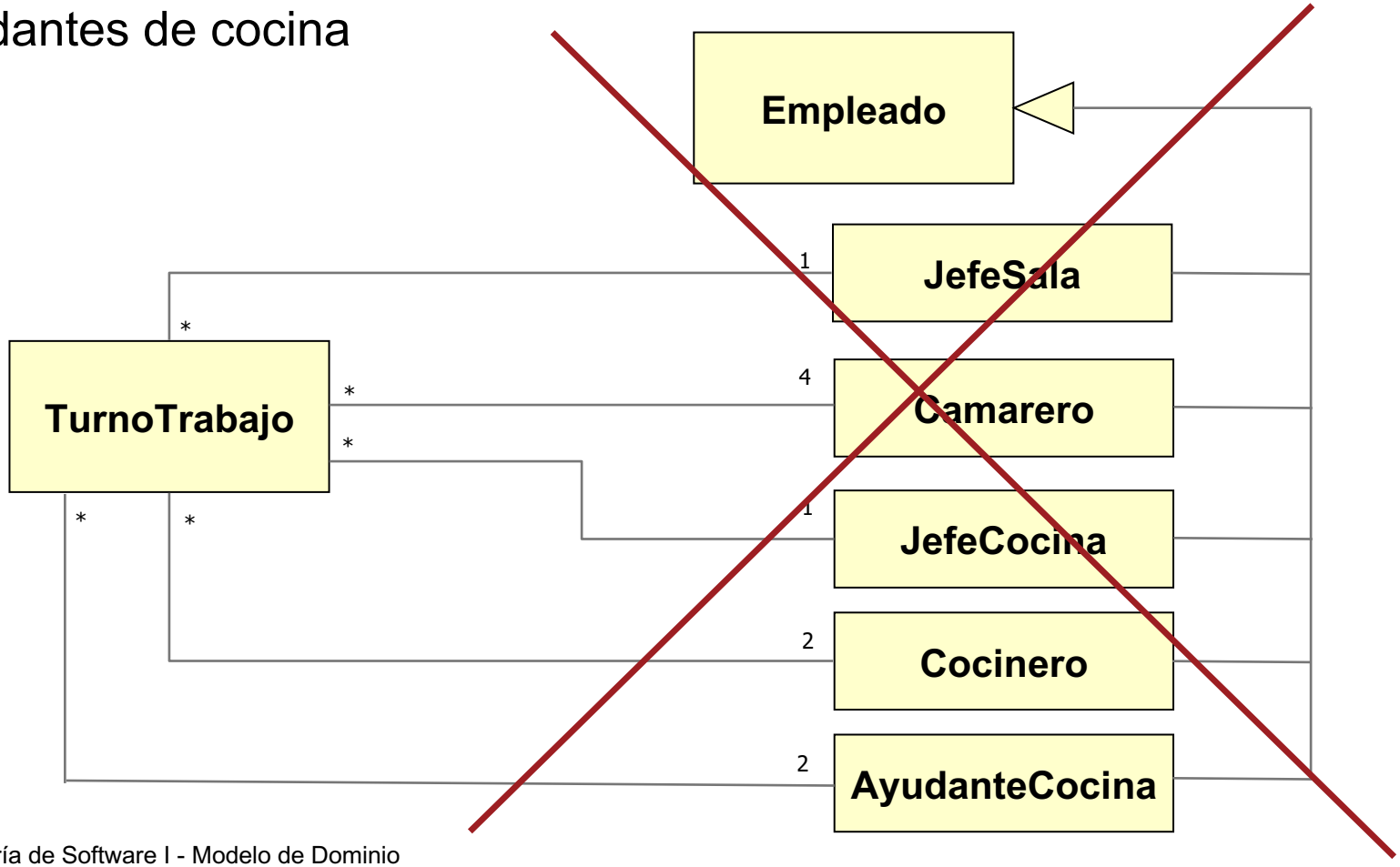
IDENTIFICACIÓN DE RELACIONES DE GENERALIZACIÓN

No se debe utilizar la relación de generalización para modelar situaciones estructurales que son susceptibles de cambiar con el paso del tiempo

- La relación de generalización es estática
- Es muy frecuente que los modelos tengan que reflejar situaciones estructurales que cambian con el paso del tiempo
- Se debe valorar modelar mediante roles

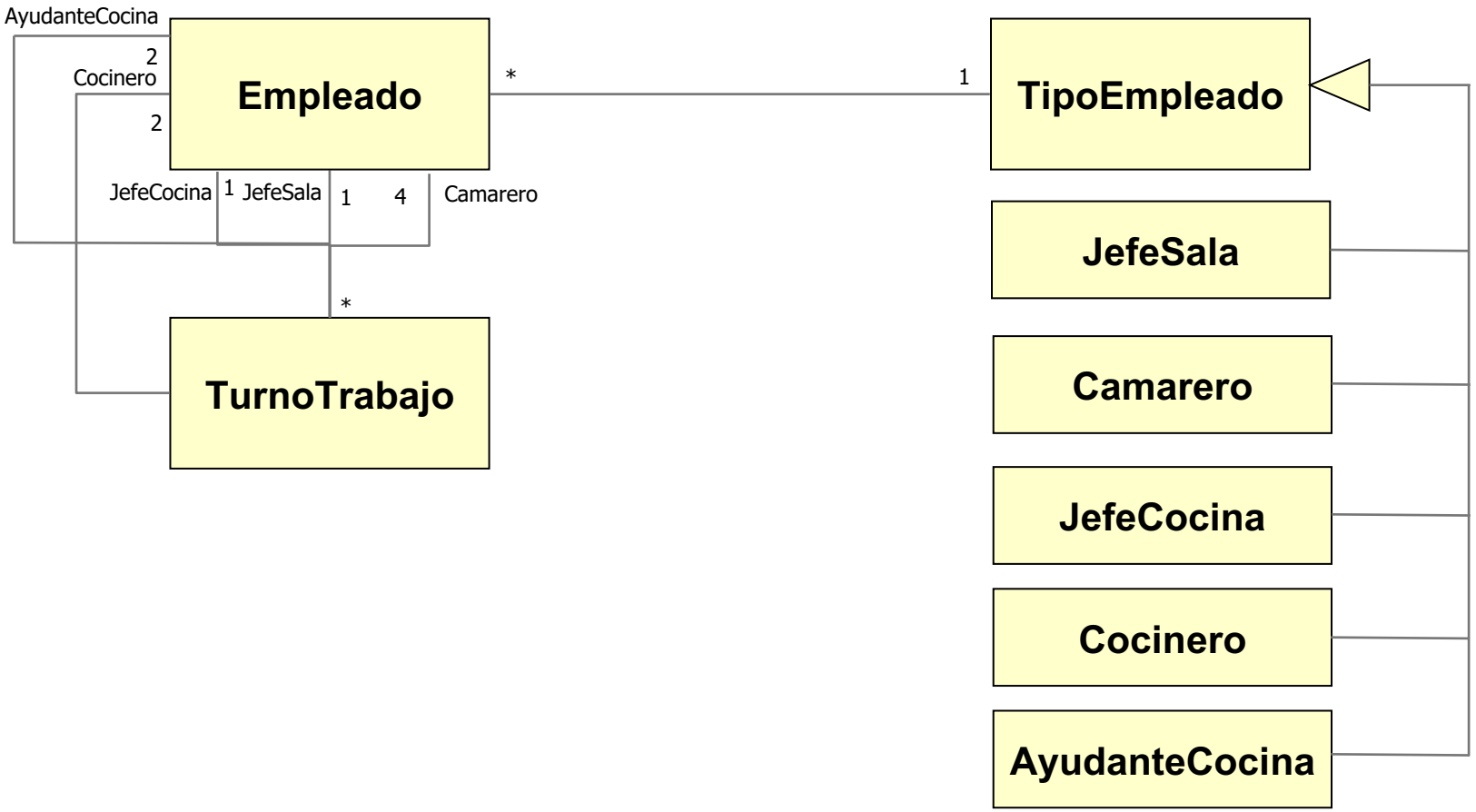
IDENTIFICACIÓN DE RELACIONES DE GENERALIZACIÓN

Un turno de trabajo en un restaurante está compuesto por un jefe de sala, cuatro camareros, un jefe de cocina, dos cocineros y dos ayudantes de cocina



IDENTIFICACIÓN DE RELACIONES DE GENERALIZACIÓN

Un turno de trabajo en un restaurante está compuesto por un jefe de sala, cuatro camareros, un jefe de cocina, dos cocineros y dos ayudantes de cocina



IDENTIFICACIÓN DE RELACIONES TODO-PARTE

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

Una relación todo-parte puede utilizarse cuando (Larman, 2004)

- 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

Agregación vs. Composición

- 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

- Agregación vs. Composición

Agregación

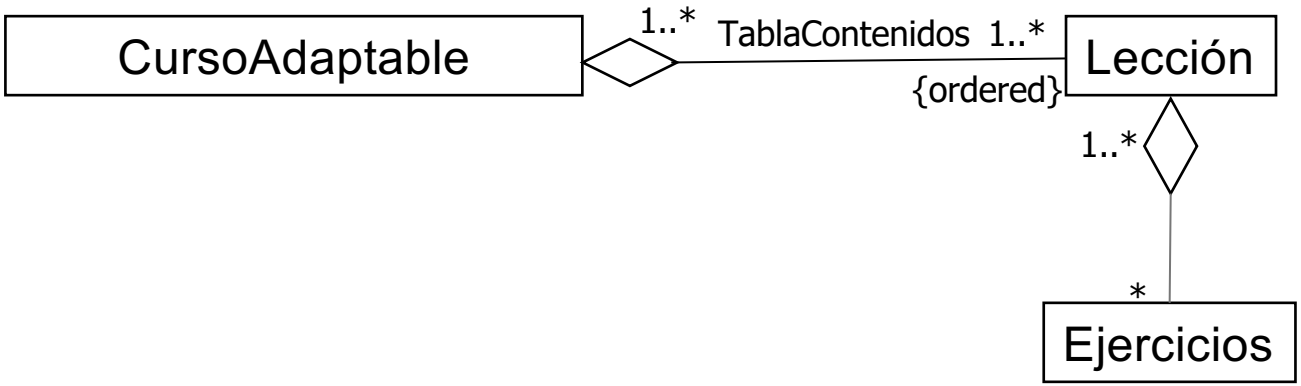
- Multiplicidad en las dos partes de la relación
- Las partes pueden existir incluso después de que el agregado sea "desmontado" o destruido
- Las partes pueden cambiar de un agregado a otro

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

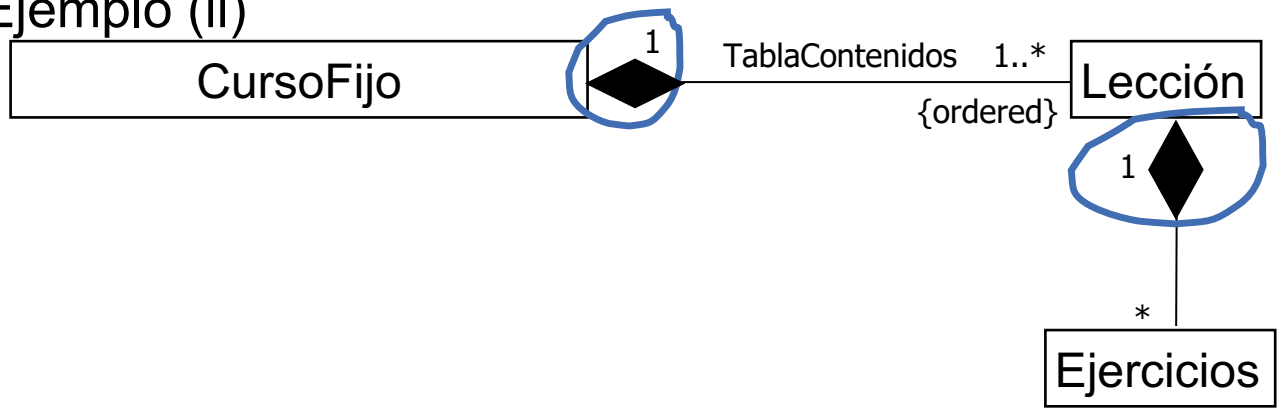
Ejemplo



- 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

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

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, 2004)

- 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

BIBLIOGRAFÍA

- P. Coad y E. Yourdon, *OOA - Object-Oriented Analysis* (Yourdon Press Computing Series). Englewood Cliffs, NJ., USA: Prentice-Hall, 1990.
- M. Fowler, *Analysis Patterns: Reusable Object Models*. Boston, USA: Addison-Wesley, 1997.
- F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, "Identificación de clases conceptuales, asociaciones y atributos," Recursos docentes de la asignatura Ingeniería de Software I. Grado en Ingeniería Informática. Curso 2020-2021, F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, Eds., Salamanca, España: Grupo GRIAL, Universidad de Salamanca, 2021a. [Online]. Disponible en: https://bit.ly/3dUDNzL. doi: 10.5281/zenodo.5784458.
- F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, "Identificación de relaciones de generalización/especialización," Recursos docentes de la asignatura Ingeniería de Software I. Grado en Ingeniería Informática. Curso 2020-2021, F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, Eds., Salamanca, España: Grupo GRIAL, Universidad de Salamanca, 2021b. [Online]. Disponible en: https://bit.ly/3GQlHLV. doi: 10.5281/zenodo.5784484.
- F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, "Identificación de relaciones todo-parte," Recursos docentes de la asignatura Ingeniería de Software I. Grado en Ingeniería Informática. Curso 2020-2021, F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, Eds., Salamanca, España: Grupo GRIAL, Universidad de Salamanca, 2021c. [Online]. Disponible en: https://bit.ly/323UVRf. doi: 10.5281/zenodo.5784490.
- F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, "Modelado de dominio," Recursos docentes de la asignatura Ingeniería de Software I. Grado en Ingeniería Informática. Curso 2020-2021, F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, Eds., Salamanca, España: Grupo GRIAL, Universidad de Salamanca, 2021d. [Online]. Disponible en: https://bit.ly/33Ch9KR. doi: 10.5281/zenodo.5784445.
- F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, "Análisis orientado a objetos," Recursos docentes de la asignatura Ingeniería de Software I. Grado en Ingeniería Informática. Curso 2023-2024, F. J. García-Peñalvo, A. García-Holgado y A. Vázquez-Ingelmo, Eds., Salamanca, España: Grupo GRIAL, Universidad de Salamanca, 2024. [Online]. Disponible en: https://d66z.short.gy/fYmTYe. doi: 10.5281/zenodo.10629829.
- D. Hay, *Data Model Patterns: Conventions of Thought*. New York, NY, USA: Dorset House, 1996.
- C. Larman, *Applying UML and patterns. An introduction to object-oriented analysis and design and the Unified Process*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2004.
- J. Martin y J. Odell, *Object-Oriented Methods: A Foundation*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1997.
- J. Rumbaugh, I. Jacobson y G. Booch, *The Unified Modeling Language reference manual*, 2ª ed. (Object Technology Series). Boston, MA, USA: Addison Wesley, 2005.
- S. Shlaer y S. J. Mellor, *Object-Oriented Analysis: Modeling the World in Data* (Yourdon Press Computing Series). Englewood Cliffs, N.J., USA: Prentice-Hall, 1988.

MODELO DE DOMINIO

INGENIERÍA DE SOFTWARE I

2º DE GRADO EN INGENIERÍA INFORMÁTICA
CURSO 2023/2024

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

Departamento de Informática y Automática
Universidad de Salamanca