Diseño Orientado a Objetos con UML

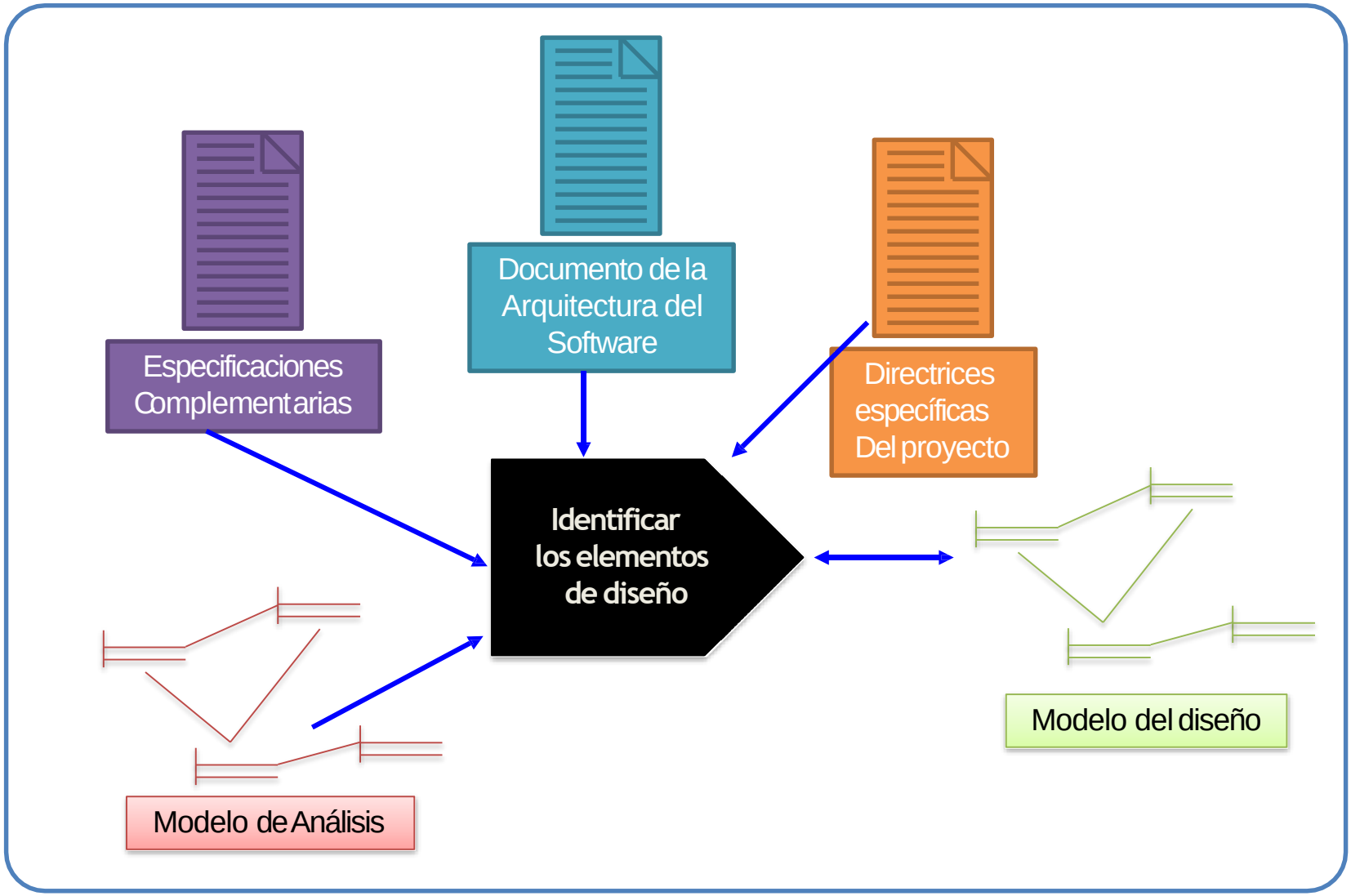
Identificando los elementos del diseño

Construyendo la arquitectura

Objetivos:
Identificar los elementos del diseño

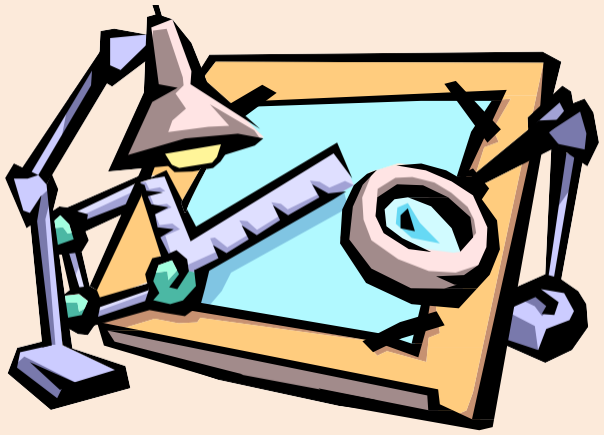
- Definir el propósito de los identificador elementos del diseño y demostrar en qué parte del ciclo de vida se lleva a cabo.
- Analizar las interacciones de las clases de análisis e identificar los elementos del diseño orientado a objetos:
  - Clases del diseño
  - Subsistemas
  - Interfaces de los subsistemas.

Identificar los elementos de diseño

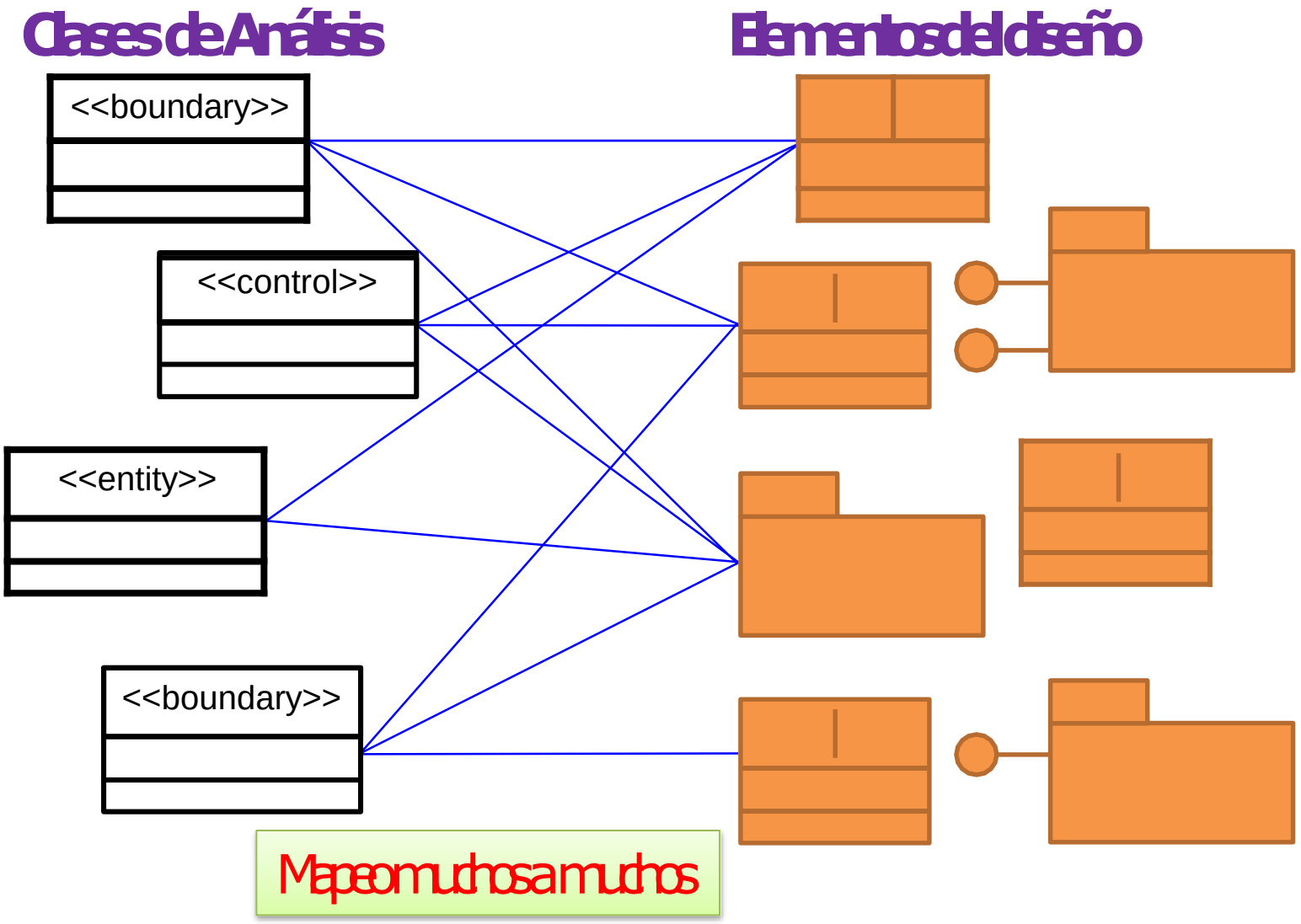


Pasos para identificar los elementos del diseño

1. Identificar las clases y subsistemas
2. Identificar las interfaces de los subsistemas
3. Actualización de la organización del Modelo de Diseño
4. Puntos de control



De las clases de análisis a los elementos del diseño

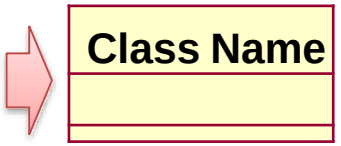


Identificando las clases de diseño

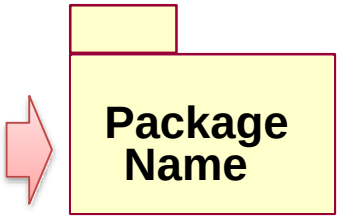
- Una clase de análisis mapea directamente a una clase de diseño si:
  - Es una clase simple
  - Representa una única abstracción lógica
- Más clases de análisis complejos pueden
  - Dividirse en múltiples clases
  - Convertirse en un paquete
  - Convertirse en un subsistema
  - Cualquier combinación ...

Repaso: Clases y Paquetes

- **¿Qué es una clase?**
  - Una descripción de un conjunto de objetos que comparten las mismas responsabilidades, relaciones, operaciones, atributos, y la semántica.

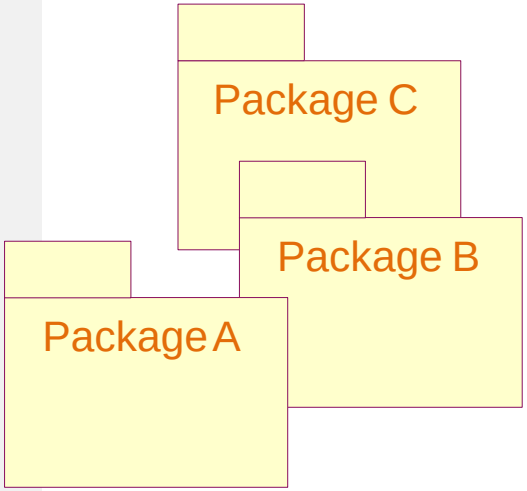


- **¿Qué es un paquete?**
  - Un mecanismo con un propósito general que agrupa a los elementos de una misma organización.
    - Un elemento del que puede modelo contener otros elementos del mismo modelo.



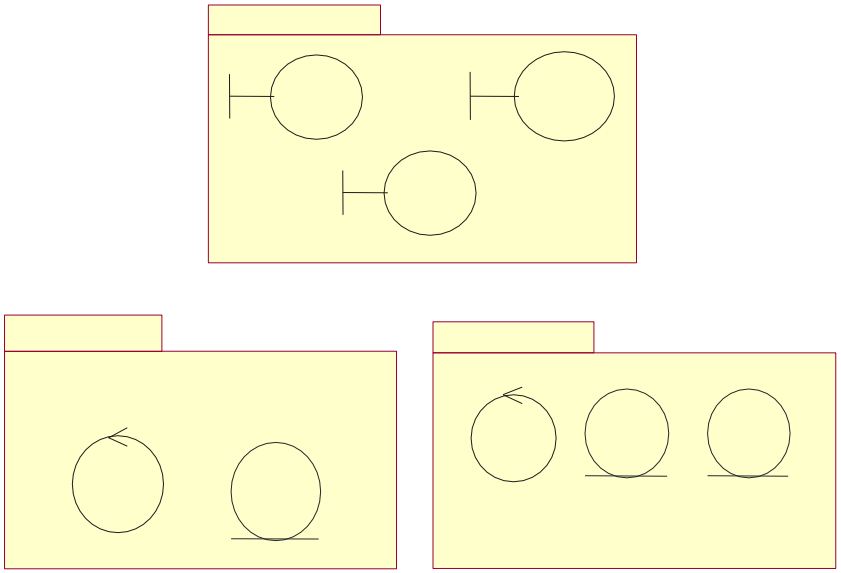
Grupo de clases de diseño en paquetes

- Puede basar sus criterios de agrupación de acuerdo a una serie de factores tales como:
  - Unidades de configuración o por tipos de clase.
  - La asignación de recursos entre los equipos de desarrollo.
  - Que refleje los tipos de usuario.
  - Representar a los productos existentes y servicios de los usos del sistema.



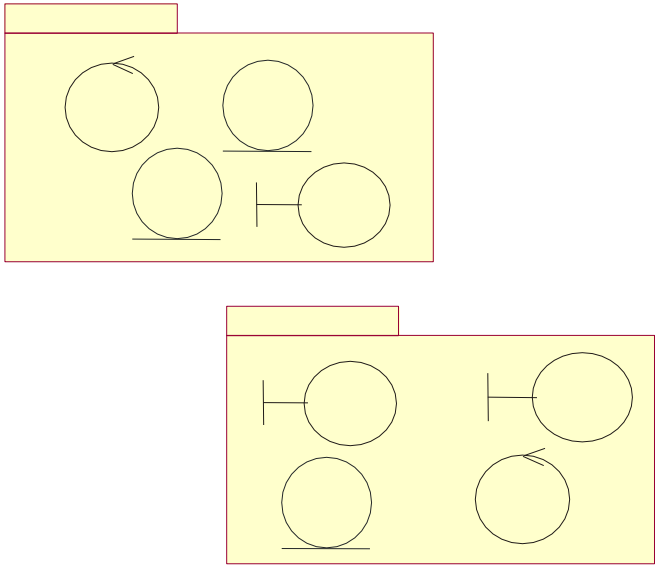
Tips para los paquetes: Clases frontera

Si existe una probabilidad considerable de que cambien las interfaces del sistema.



Colocar las clases frontera en paquetes separados.

Si es poco probable que la interfaz del sufrirá cambios considerables



Empaquetar con clases funcionalmente relacionados

Consejos para los paquetes: Clases funcionalmente relacionados

Criterios para determinar si las clases están funcionalmente relacionadas:

- ✓ Los cambios en una clase de comportamiento y / o estructura necesitan cambios en otra clase.
- ✓ La eliminación de una clase impacta a otra clase.
- ✓ Dos objetos interactúan con un gran número de mensajes o tienen una intercomunicación compleja.
- ✓ Una clase frontera puede estar funcionalmente relacionada a una clase entidad en particular.
- ✓ Si la función de la clase frontera es presentar la clase de entidad.
- ✓ Dos clases interactúan con o se ven afectados por los mismos cambios.

Consejos para los paquetes: Clases funcionalmente relacionados…cont

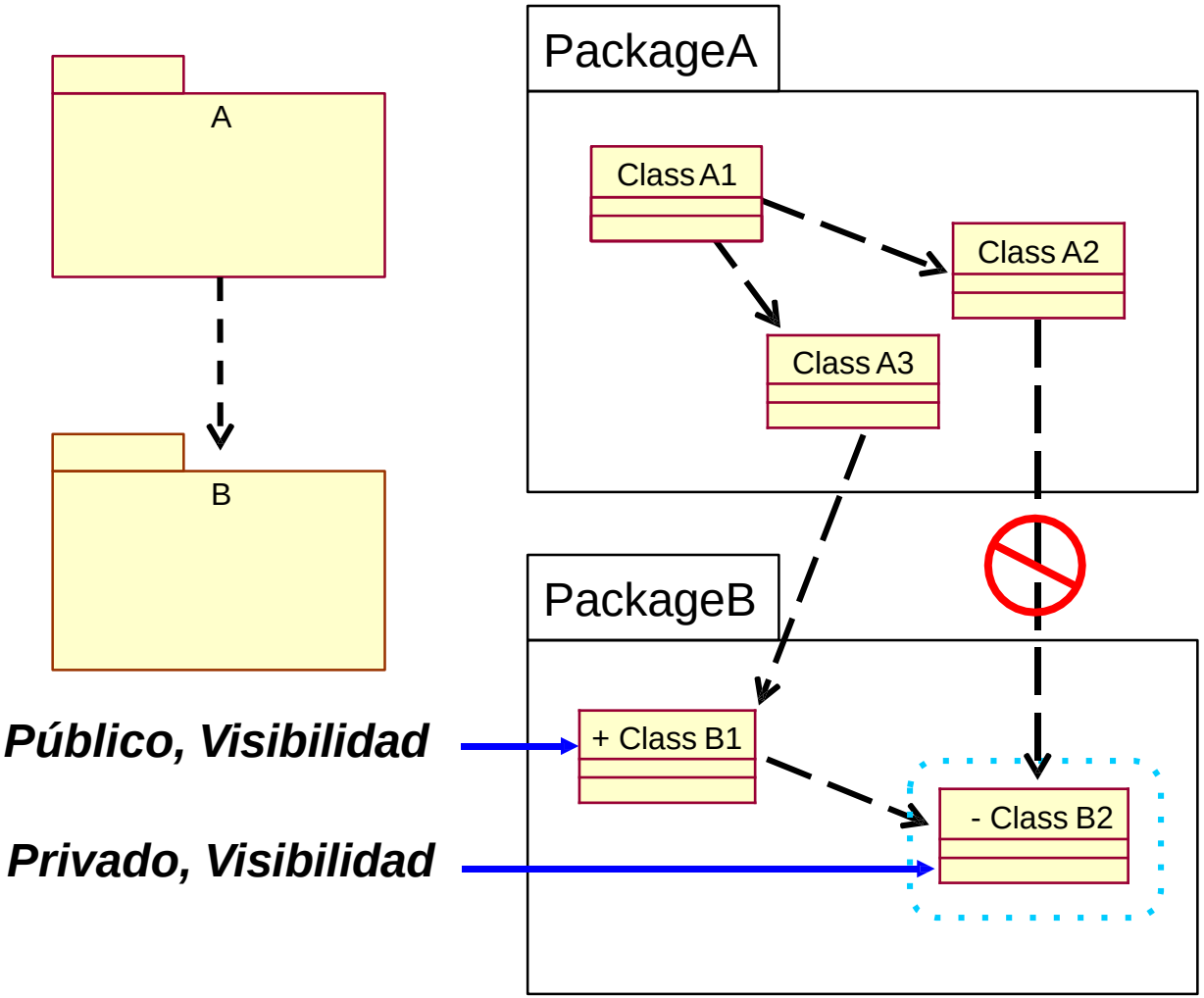
Criterios para determinar si las clases están funcionalmente relacionadas:

- ✓ Dos clases tienen relaciones entre sí
- ✓ Una clase crea instancias de otra clase

Criterios para determinar cuando dos clases no deben ser colocados en el mismo paquete:

- ✓ Dos clases que están relacionados con los diferentes actores no deben ser colocados en el mismo paquete
- ✓ Una clase opcional y una obligatoria no debe ser colocado en el mismo paquete.

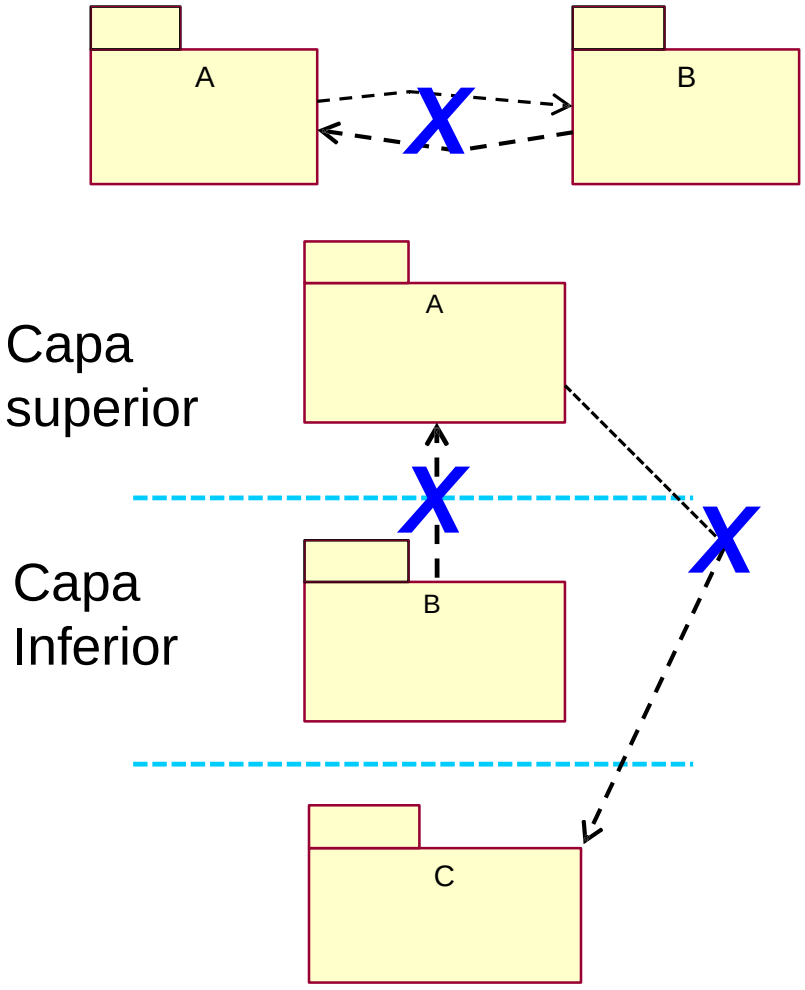
Dependencias en paquetes: Visibilidad de los elementos de un paquete



Sólo las clases públicas se puede hacer referencia exterior del paquete propietaria

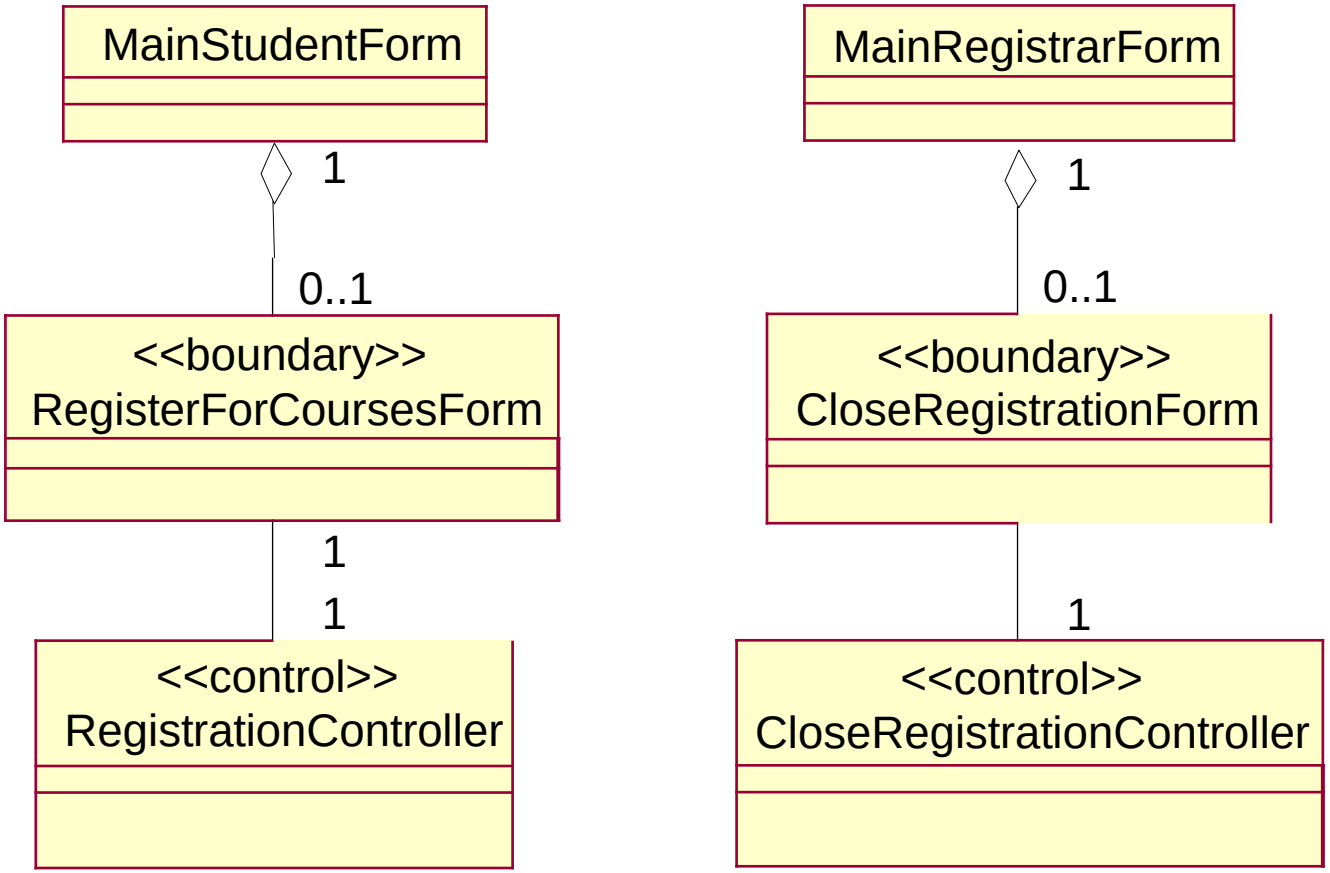
Tips para el acoplamiento de paquetes

- ✓ Los paquetes no deben tener acoplamiento cruzado.
- ✓ Los paquetes en capas inferiores no deben depender de paquetes en capas superiores.
- ✓ En general, las dependencias no deben omitir las capas.

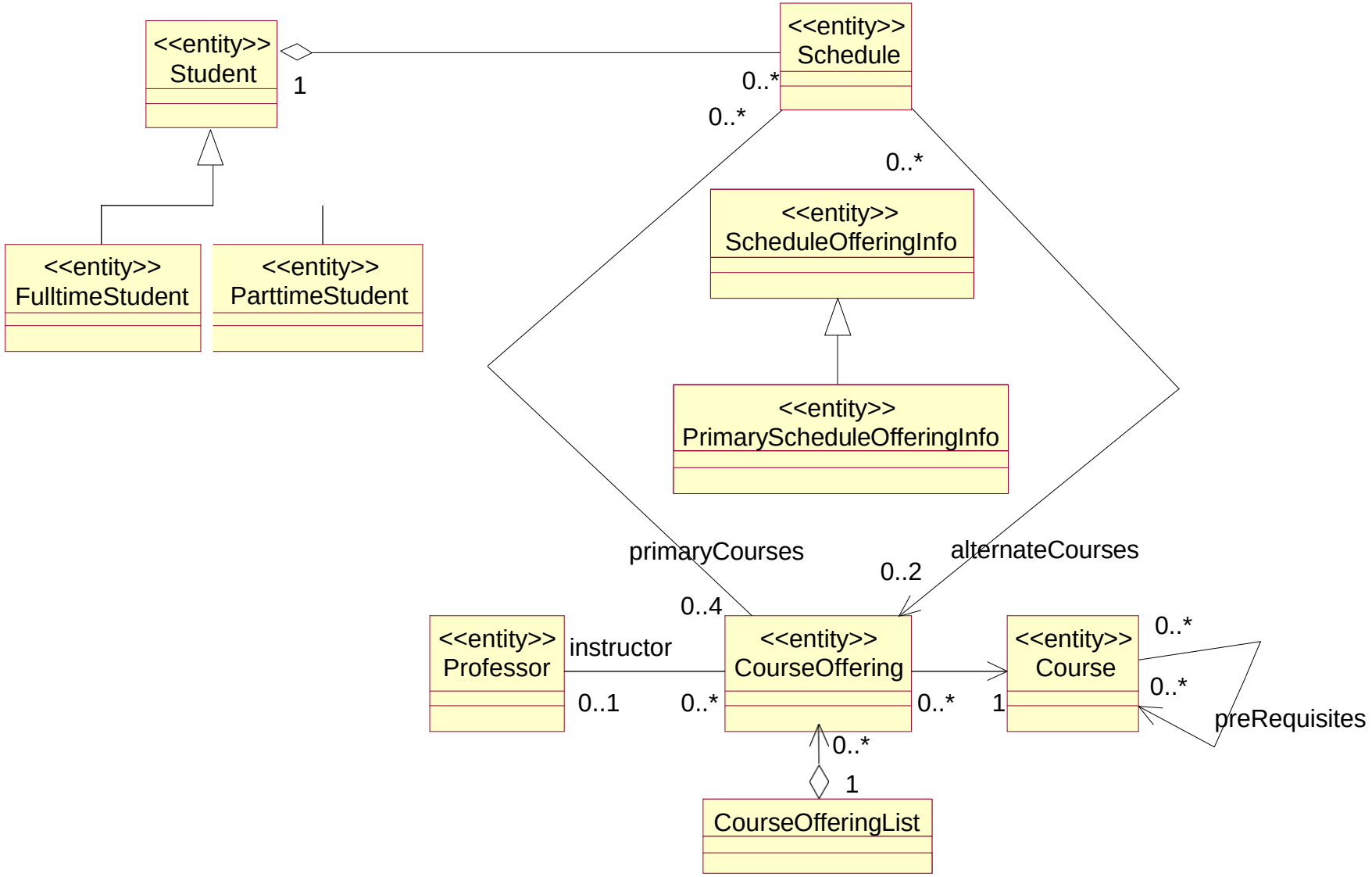


X = Violación de acoplamiento

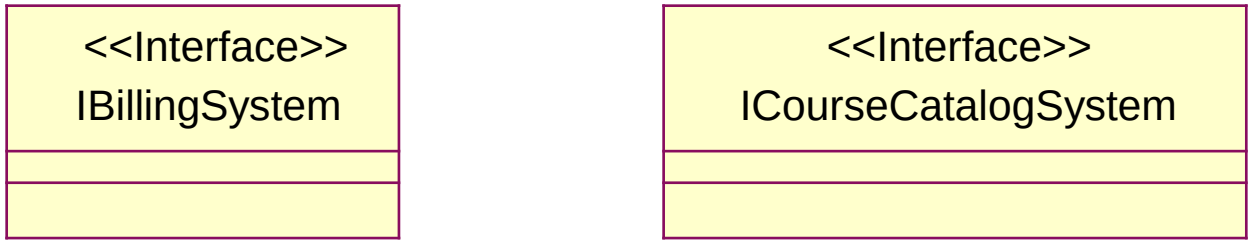
Ejemplo: Paquete de matrícula (registro)



Ejemplo: Paquetes de artefactos de una Universidad

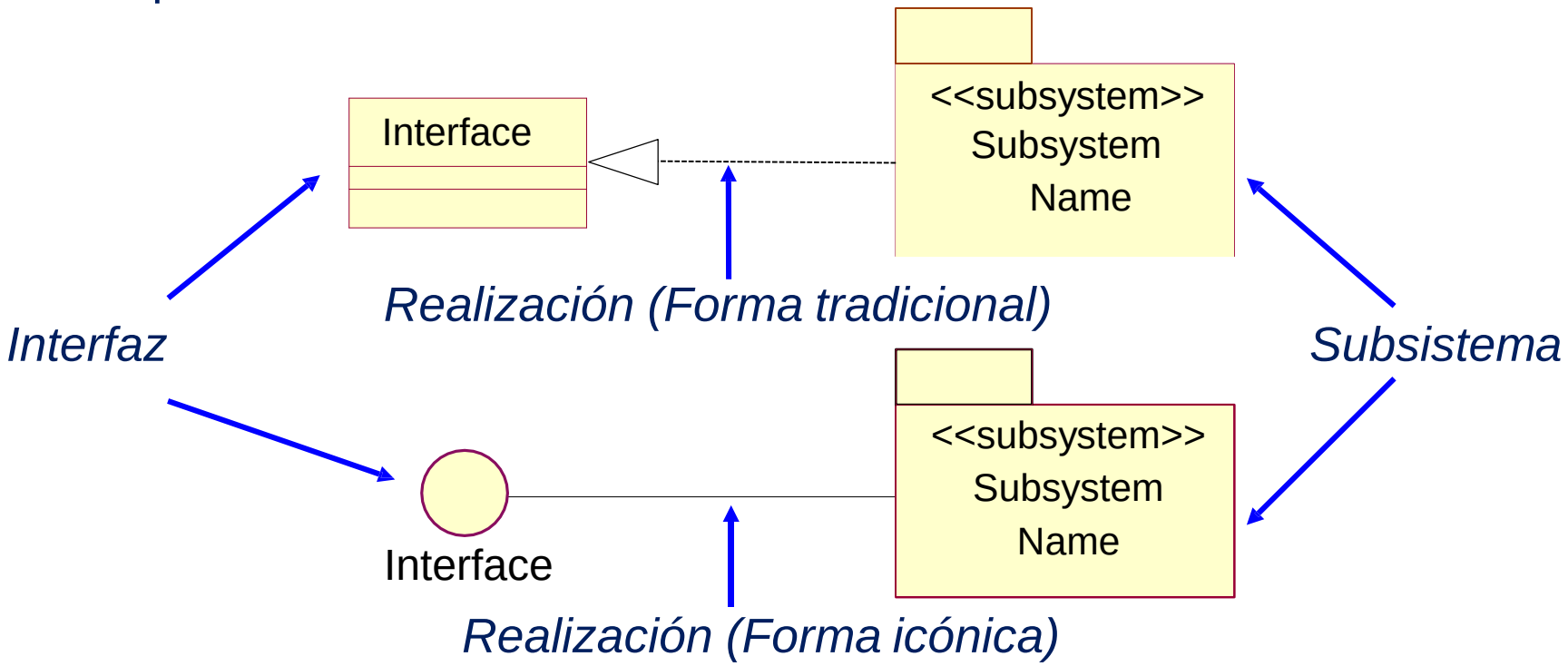


Paquete Interfaces externas de un sistema



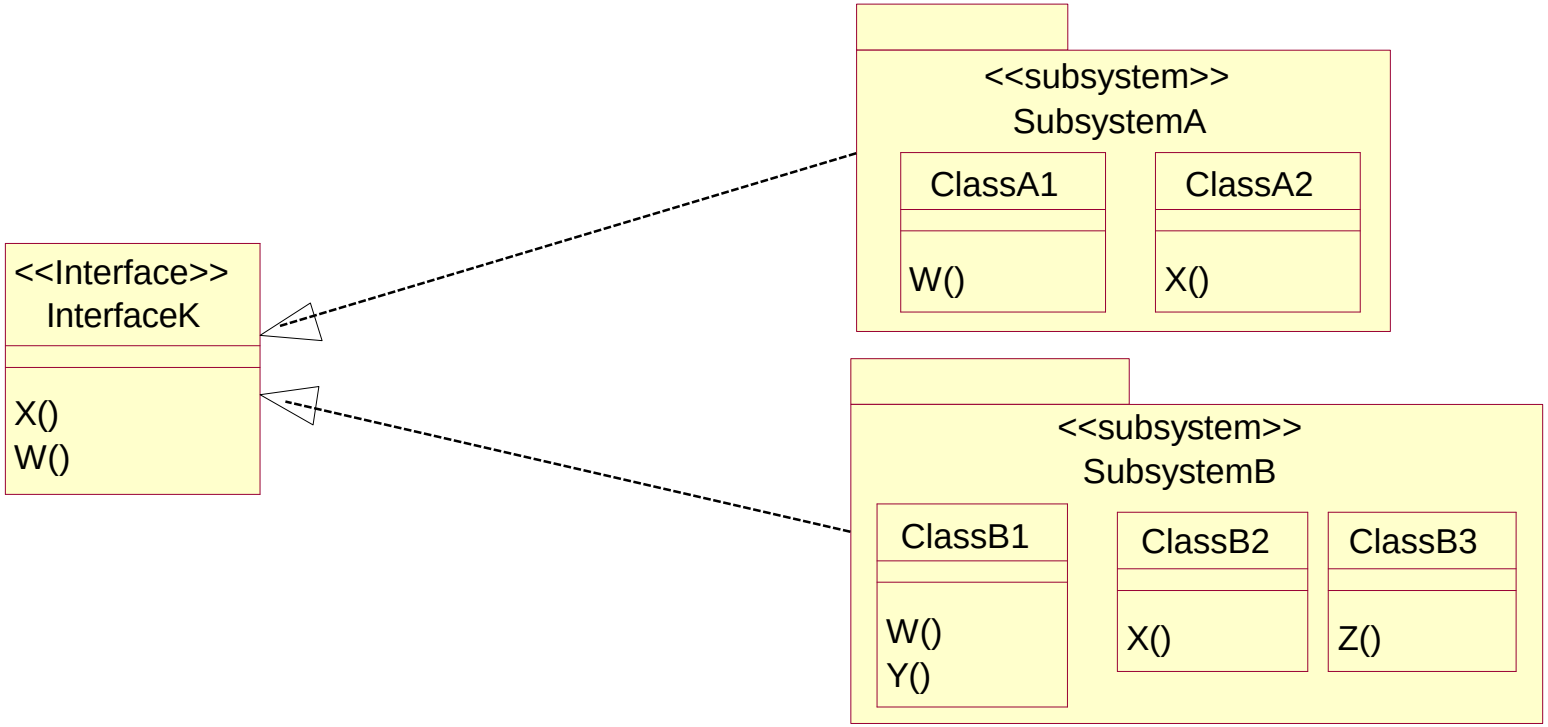
Repaso: Subsistemas e interfaces

- Son una "mezcla entre" un paquete (puede contener otros elementos del modelo) y una clase (tiene un comportamiento)
- Se accede a ellos a través de una o más interfaces que definen su comportamiento



Repaso: Subsistemas e interfaces

- Subsistemas:
  - Comportamiento completamente encapsulado.
  - Representa una capacidad independiente y con interfaces claras (potencial para la reutilización)
  - Múltiples variantes de aplicación de modelos



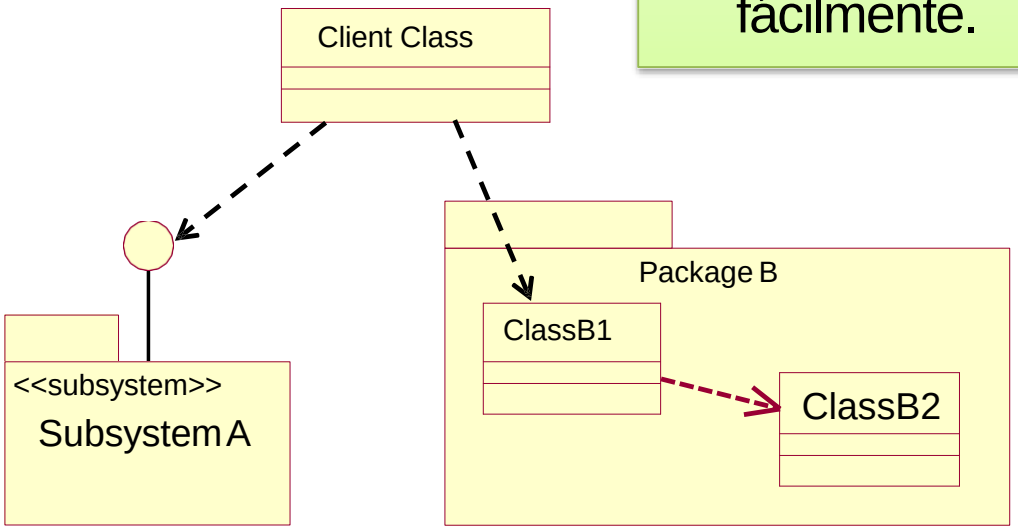
Paquetas versus Subsistemas

Subsistemas

- Proporcionan comportamiento
- Encapsulan completamente su contenido
- Son fácilmente reemplazados

Paquetes

- No proporciona comportamiento
- No encapsula completamente su contenido
- No puede ser reemplazado fácilmente.



Encapsulamiento es la clave

Uso de los subsistemas

- Los subsistemas pueden ser utilizados para dividir el sistema en partes que pueden ser independientes:
  - Ordenado, configurado, o entregado
  - Desarrollado, siempre y cuando las interfaces permanezcan sin cambios
  - Desplegado a través de un conjunto de nodos computacionales distribuidos.
  - Cambiado sin romper otras partes de los sistemas
- Subsistemas también se pueden utilizar para:
  - Dividir el sistema en unidades que pueden proporcionar seguridad restringida por recursos claves
  - Representar los productos existentes o sistemas externos en el diseño (por ejemplo, componentes)

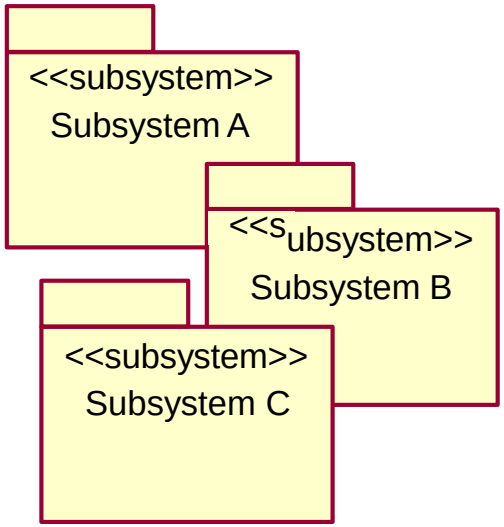
Los subsistemas elevan los niveles de abstracción

Sugerencias para identificar subsistemas

- Observe colaboraciones entre objetos.
- Busque opciones.
- Observe la interfaz de usuario del sistema.
- Observe a los actores.
- Busque acoplamiento y cohesión entre las clases.
- Observe que se puede sustituir
- Observe que puede distribuirse
- Observe volatilidad.

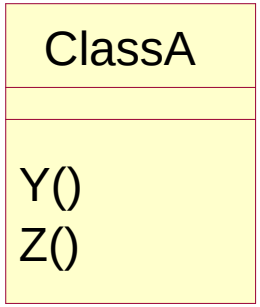
Subsistemas candidatos

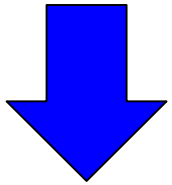
- Clases de análisis que se pueden desarrollar en subsistemas:
  - ✓ Las clases que ofrecen servicios y / o utilidades complejas
  - ✓ Clases frontera (interfaces de usuario e interfaces de sistemas externos)
- Productos o sistemas externos existentes en el diseño (por ejemplo, componentes):
  - ✓ Software de comunicación
  - ✓ Soporte de acceso a base de datos
  - ✓ Tipos y estructuras de datos
  - ✓ Utilidades comunes
  - ✓ Productos para aplicaciones específicas

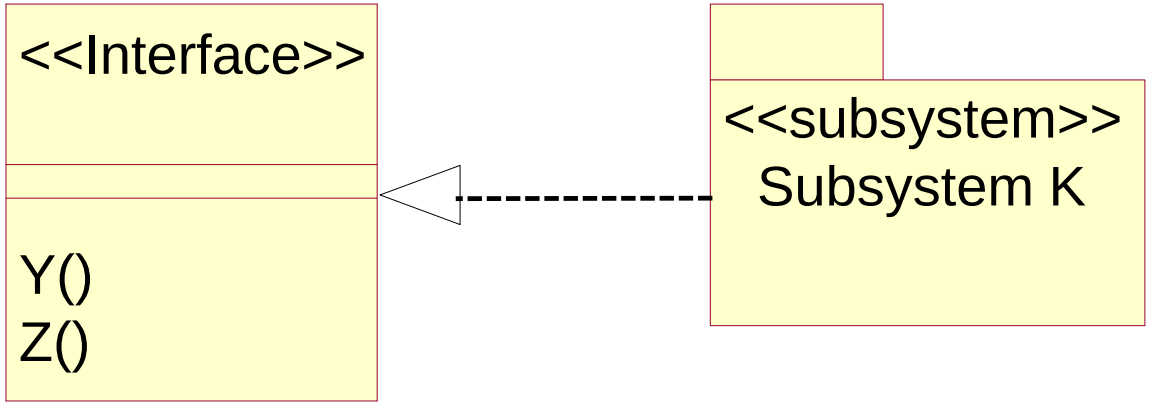


Identificando Subsistemas

"Superman Class"







Identificando Interfaces

- Propósito
  - Identificar las interfaces de los subsistemas en base a sus responsabilidades.
- Pasos
  - Identificar un conjunto de interfaces candidatos para todos los subsistemas.
  - Puedes buscar similitudes entre interfaces.
  - Definir dependencias de interfaz.
  - En el mapa los interfaces con los subsistemas.
  - Definir el comportamiento especificado por las interfaces.
  - Paquete de las interfaces.

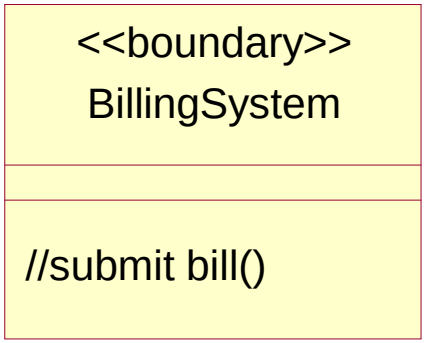
Las interfaces estables y bien definidos son clave para una arquitectura resistente estable.

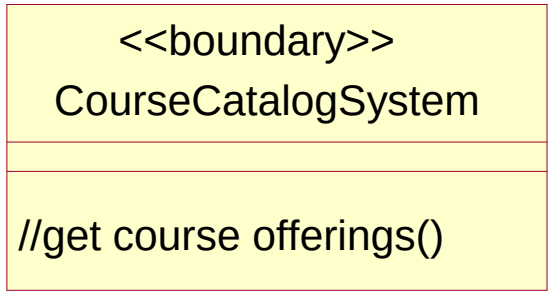
Directrices para las interfaces

- Nombre de la interfaz
  - Refleja el papel en el sistema
- Descripción de las interfaces
  - Transmite responsabilidades
- Definición de operación
  - Nombre debe reflejar resultado de la operación
  - Describe lo que hace la operación, todos los parámetros y resultados
- Documentación de la Interfaz
  - Información del paquete de soporte: secuencia y diagramas de estado, los planes de prueba, etc.

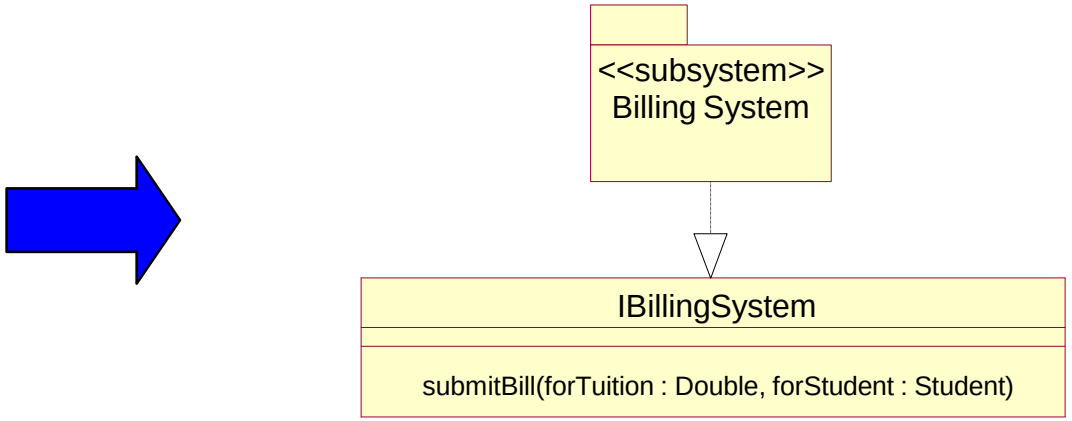
Ejemplo: Diseño de subsistemas e interfaces

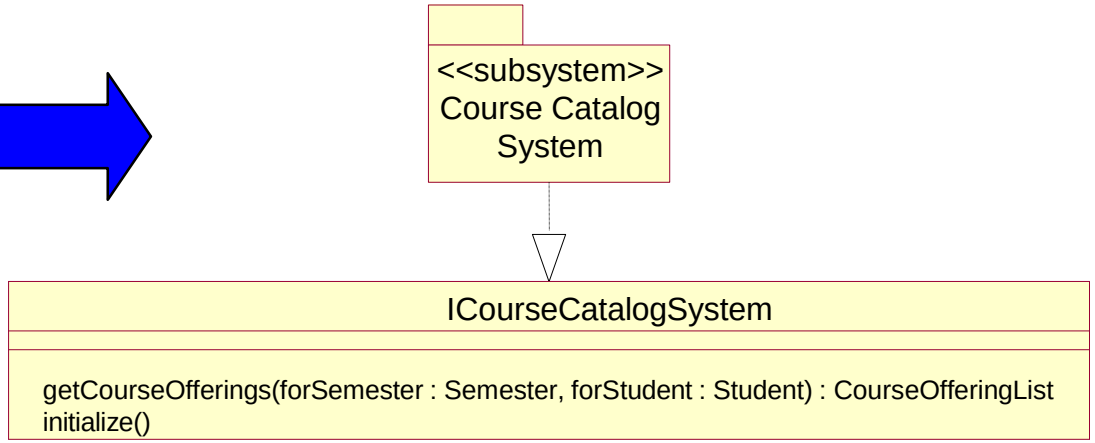
Análisis





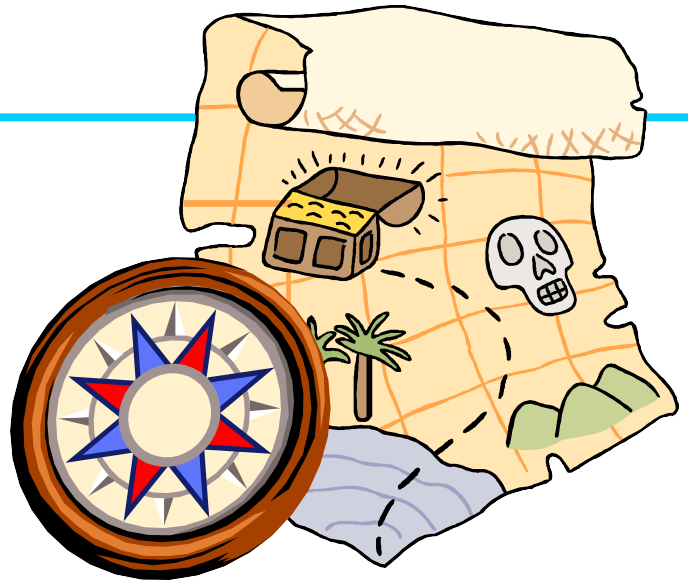
Diseño



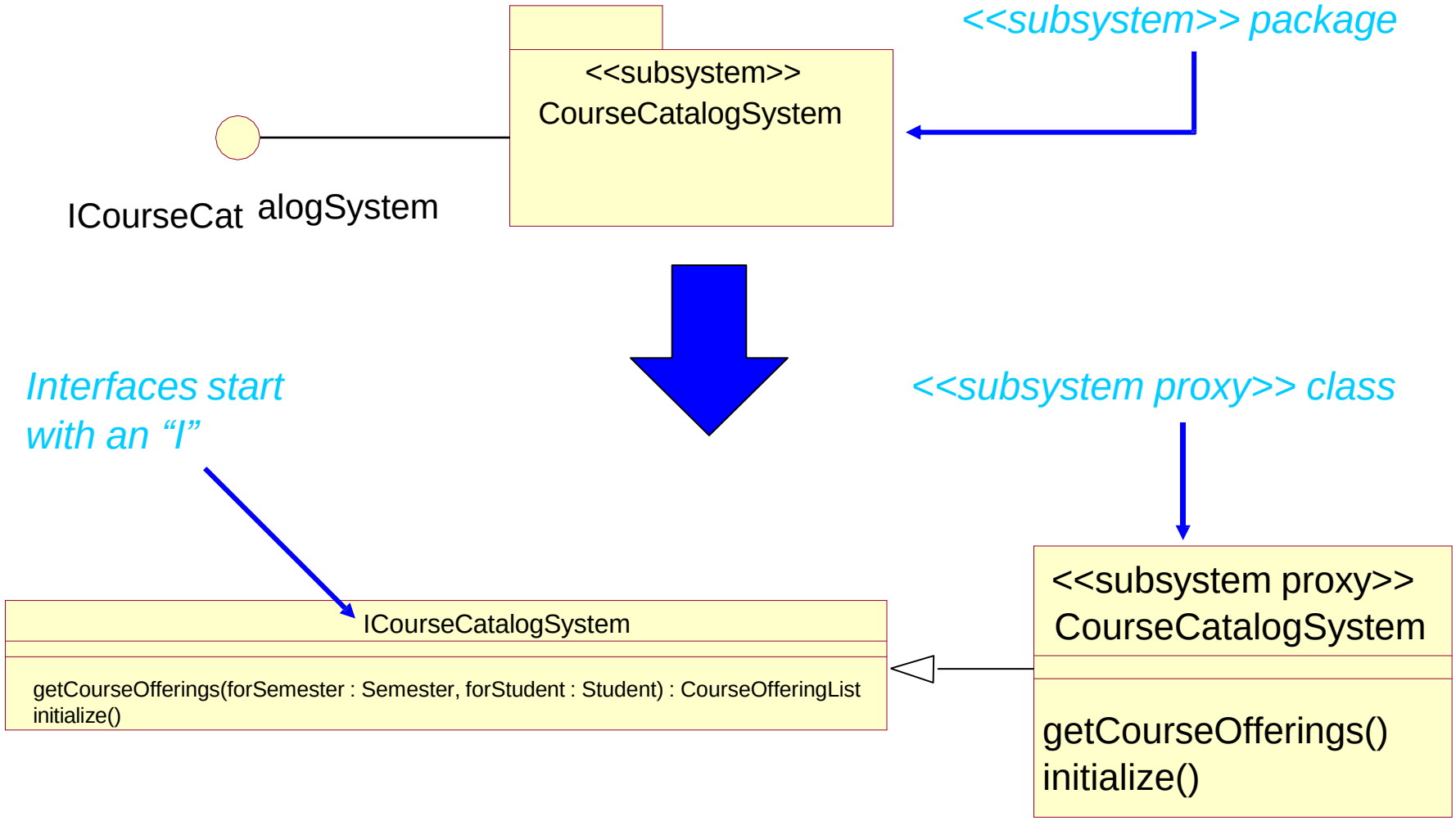


De las clases de análisis a las clases de diseño

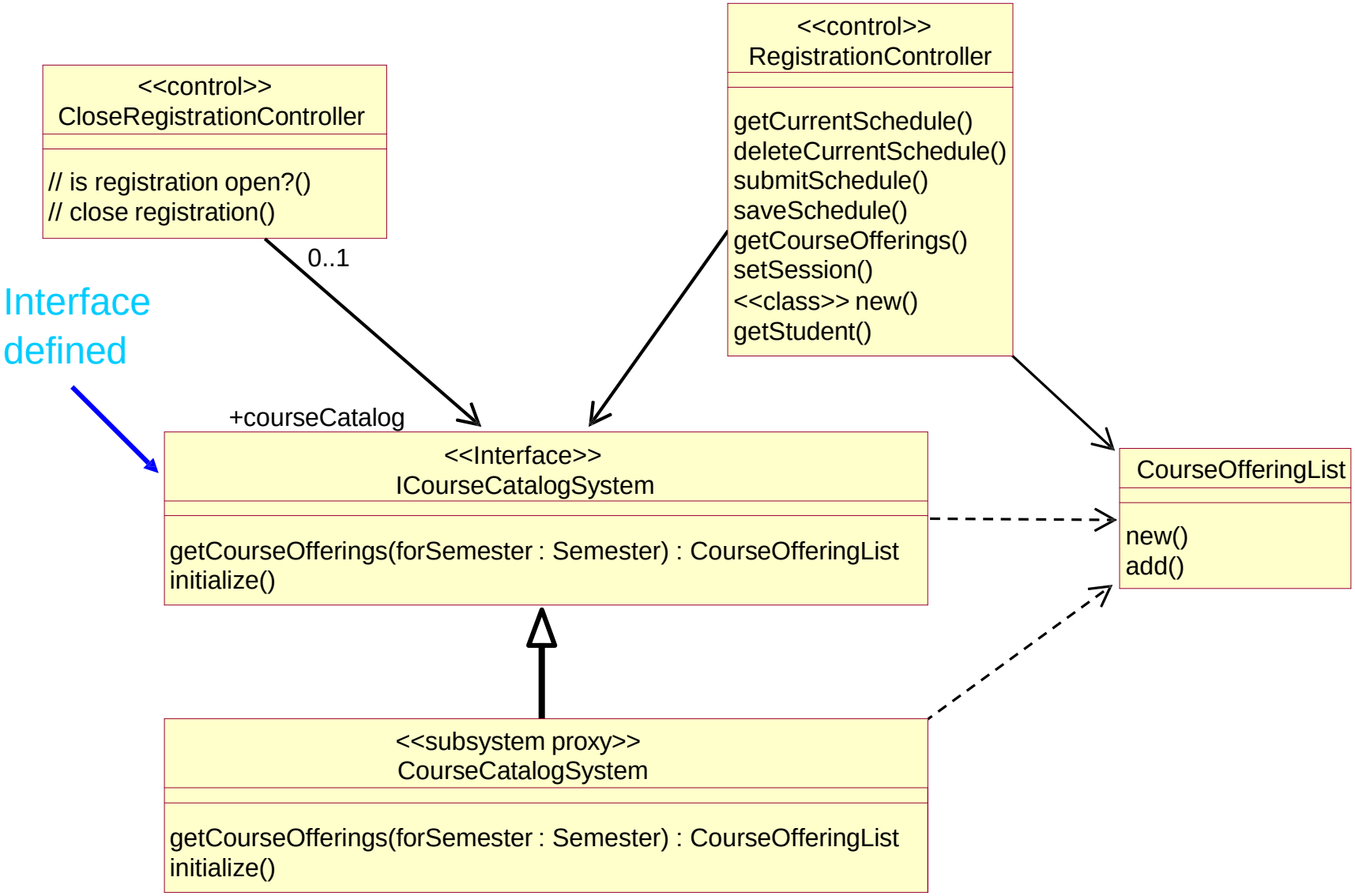
Clases de análisis	Elementos del diseño
CourseCatalogSystem	CourseCatalogSystem Subsystem
BillingSystem	BillingSystem Subsystem
Otras clases de análisis	



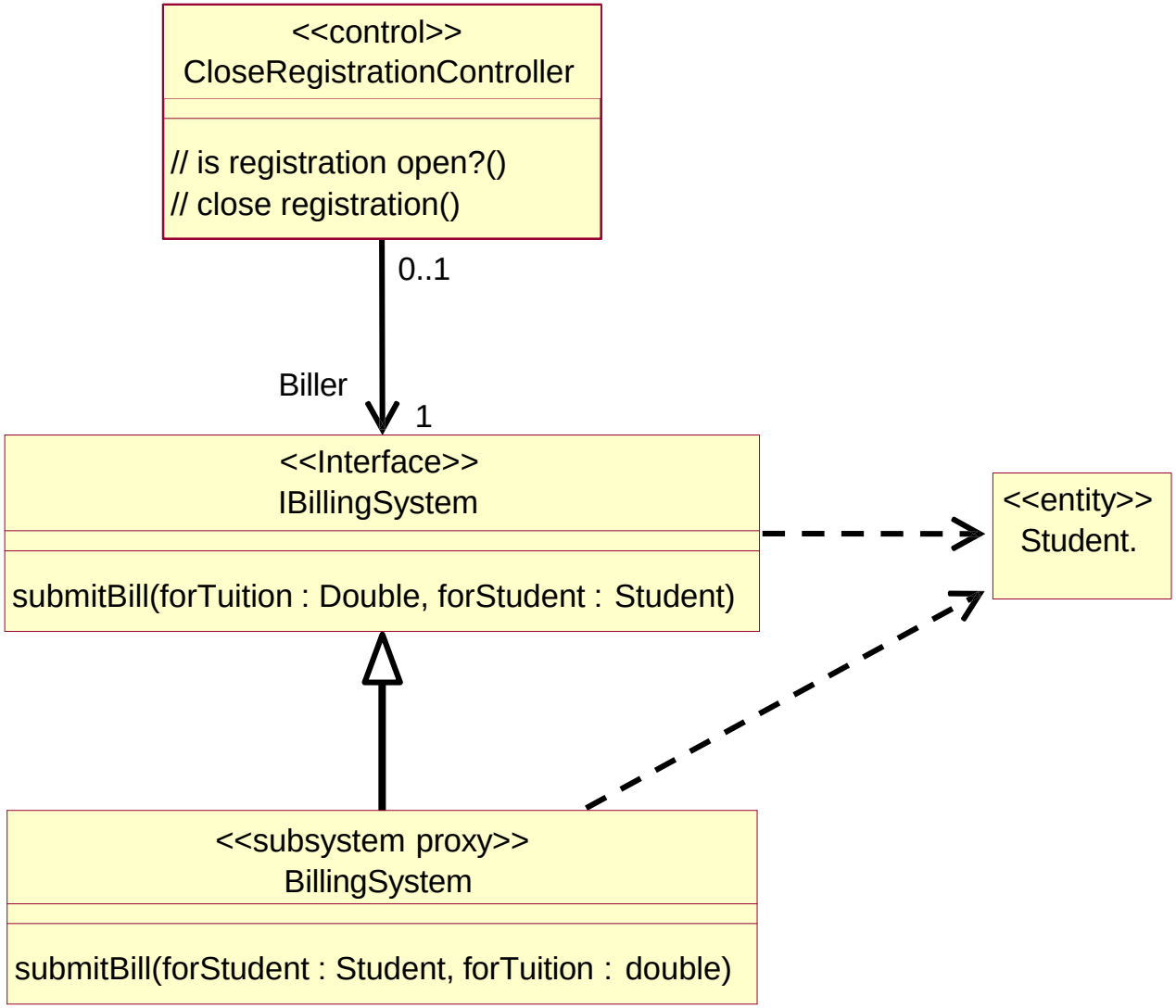
Subsistemas e interfaces



Ejemplo: Contexto Subsistema: CourseCatalogSystem



Example: Subsystem Context: Billing System

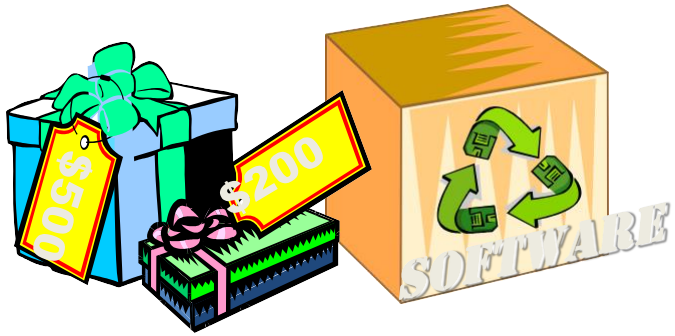


Identificación de oportunidades de reutilización

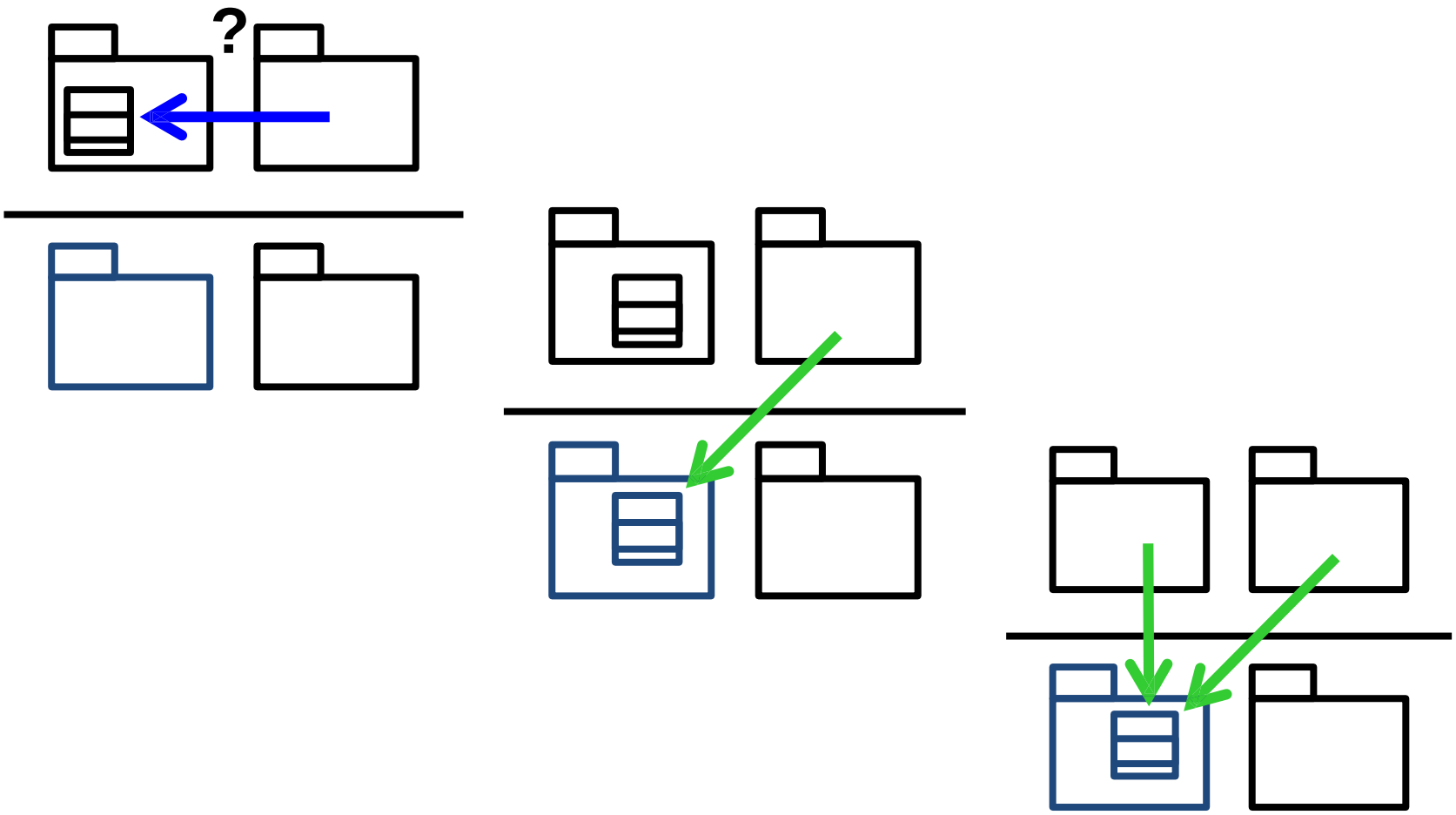
- Propósito
  - Identificar dónde puede haber subsistemas y / o componentes reutilizados en función de sus interfaces.
- Pasos
  - Busque interfaces similares
  - Modificar nuevas interfaces para mejorar los ajustes.
  - Reemplace las interfaces candidatos con interfaces existentes
  - Mapear el subsistema candidato a componentes existentes

Posibles oportunidades de reuso

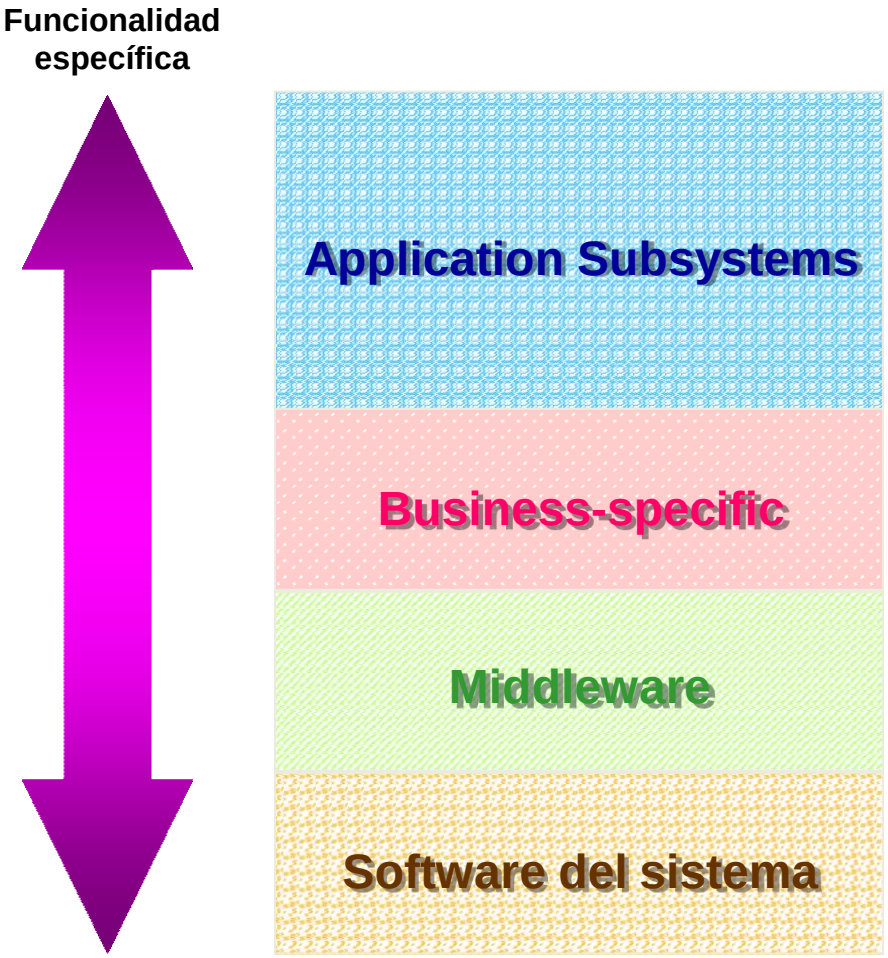
- Internas al sistema que se desarrolló
  - Comúnmente reconocido a través de paquetes y subsistemas
- Externas al sistema que se desarrolló
  - Componentes disponibles comercialmente
  - Componentes de una aplicación desarrollada previamente
  - Invierta componentes de ingeniería



Oportunidades de reutilización interna de un Sistema



Repaso: Enfoque típico por capas



Componen la aplicación, contiene el software de valor agregado desarrollado por la organización.

Contiene una serie de subsistemas reutilizables y específicos para el tipo de negocio de la empresa.

Middleware , ofrece subsistemas para las clases y servicios independientes de la plataforma de computación distribuida en objetos en entornos heterogéneos.

El software del sistema , contiene el software para la infraestructura actual, tales como sistemas operativos, interfaces con hardware específico, controladores de dispositivos entre otros

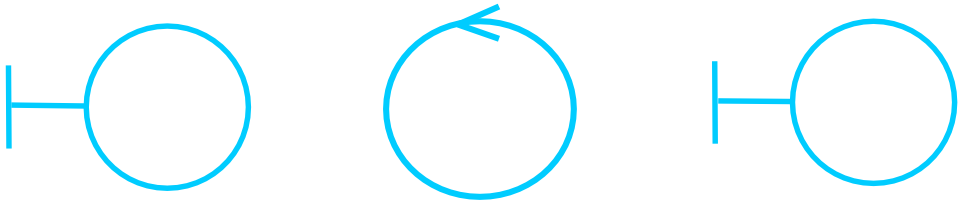
Consideraciones para trabajar en capas

- **Visibilidad**
  - Dependencias sólo dentro de capa actual y por debajo
- **Volatilidad**
  - Las capas superiores afectados por cambios en los requerimientos
  - Capas inferiores afectados por cambios en el entorno
- **Generalidad**
  - Más elementos de modelos abstractos en las capas inferiores
- **Número de capas**
  - Pequeño sistema: 3-4 capas
  - Sistema complejo: 5-7 capa

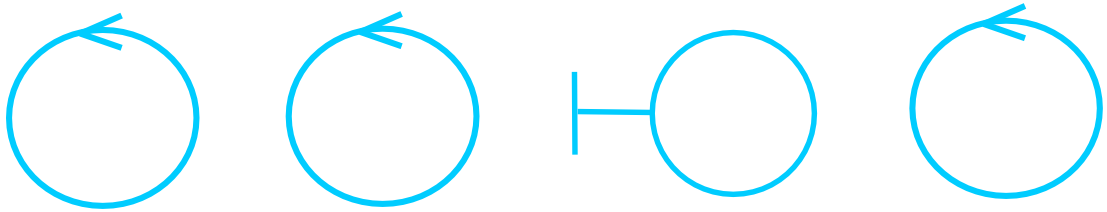
El objetivo es reducir el acoplamiento para aliviar el esfuerzo de mantenimiento.

Elementos de diseño y la arquitectura

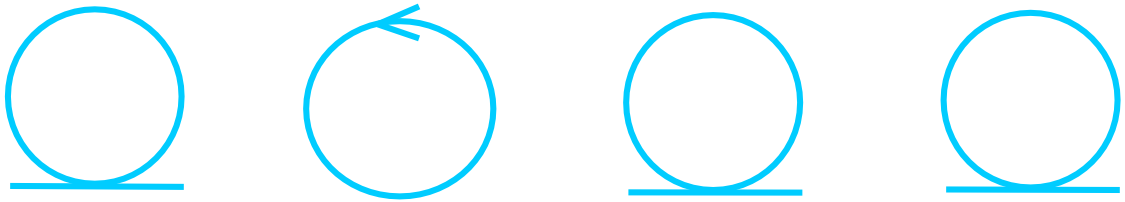
Capa 1



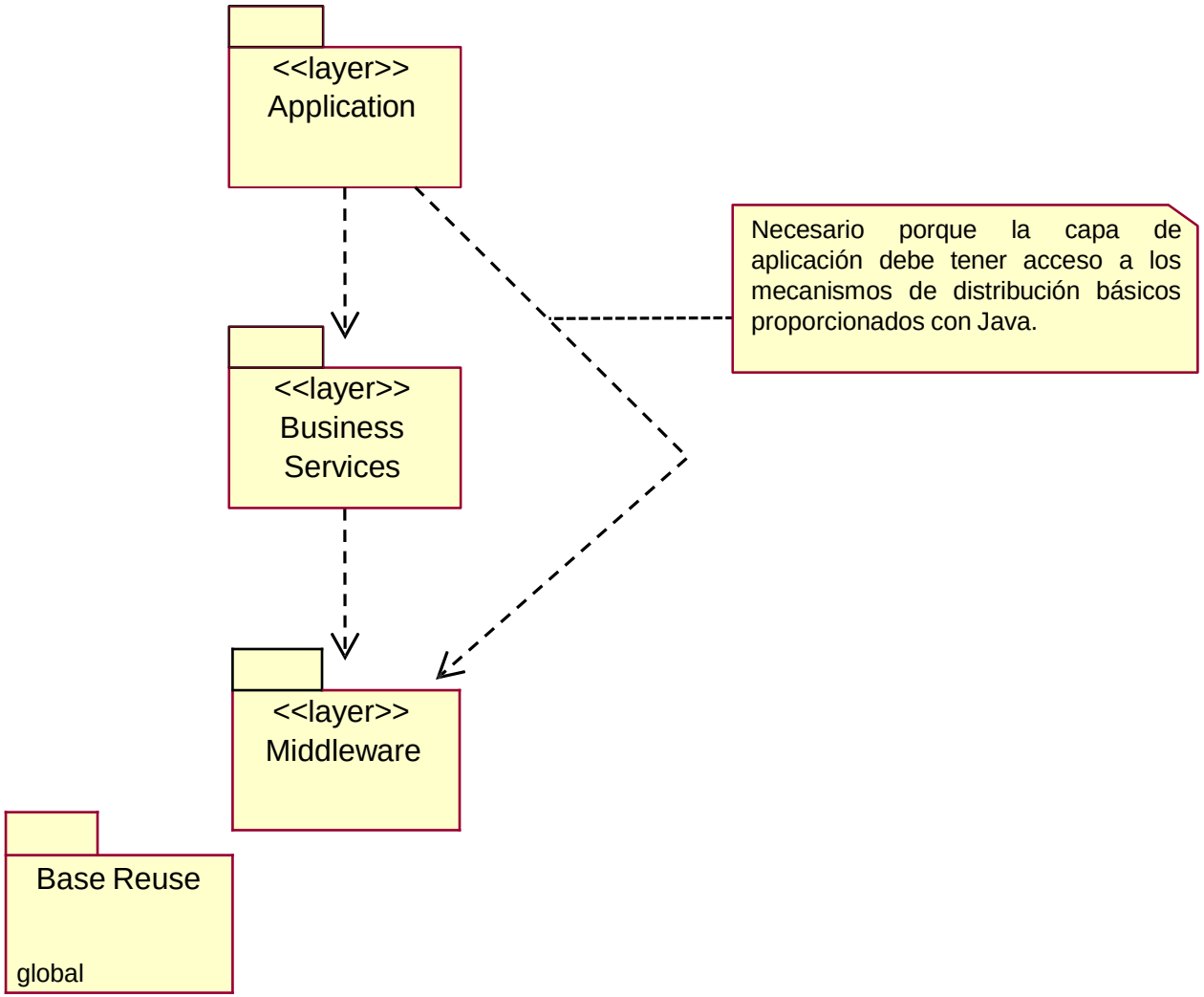
Capa 2



Capa 3



Ejemplo: Las capas de arquitectura

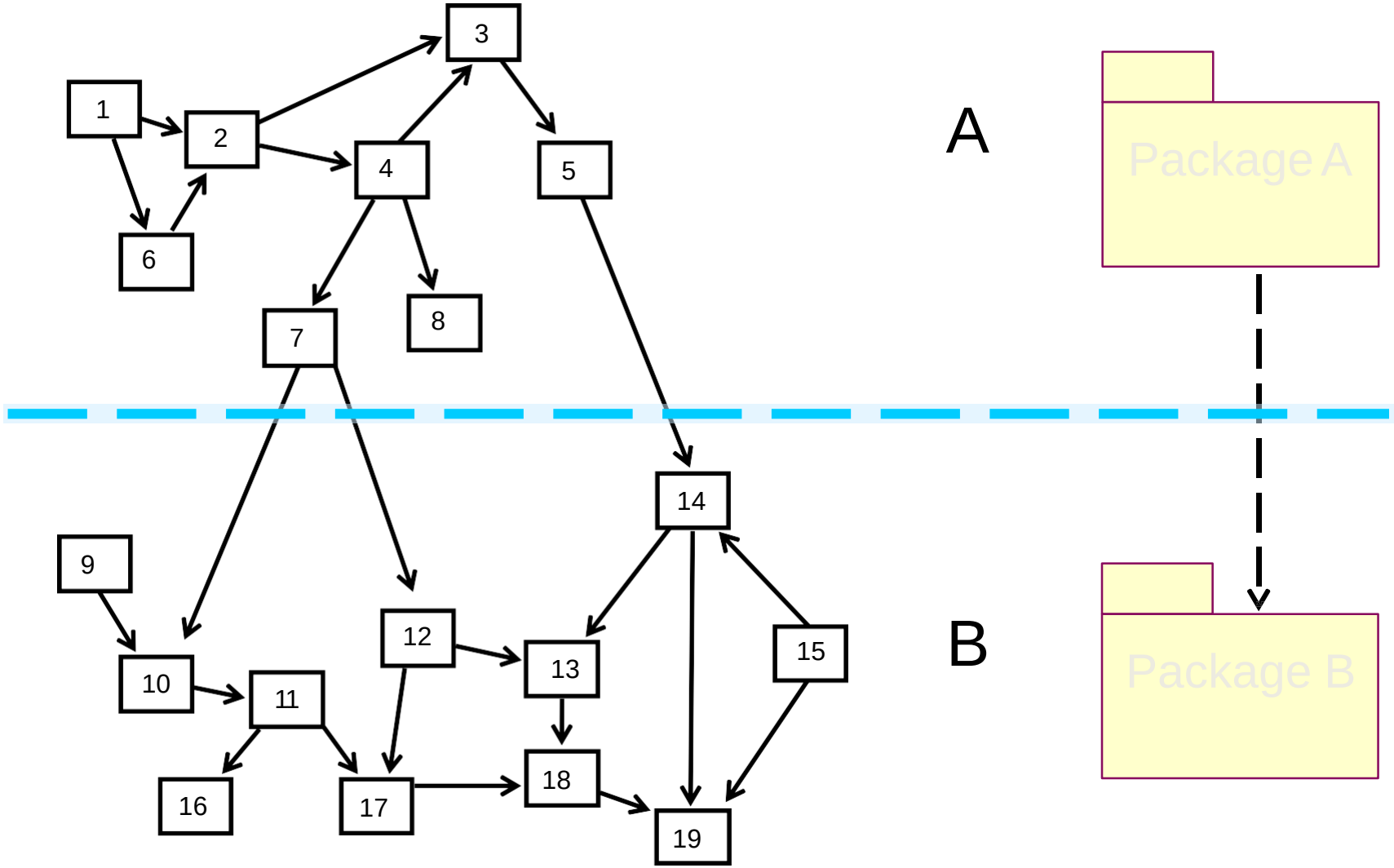


Consideraciones de particionamiento

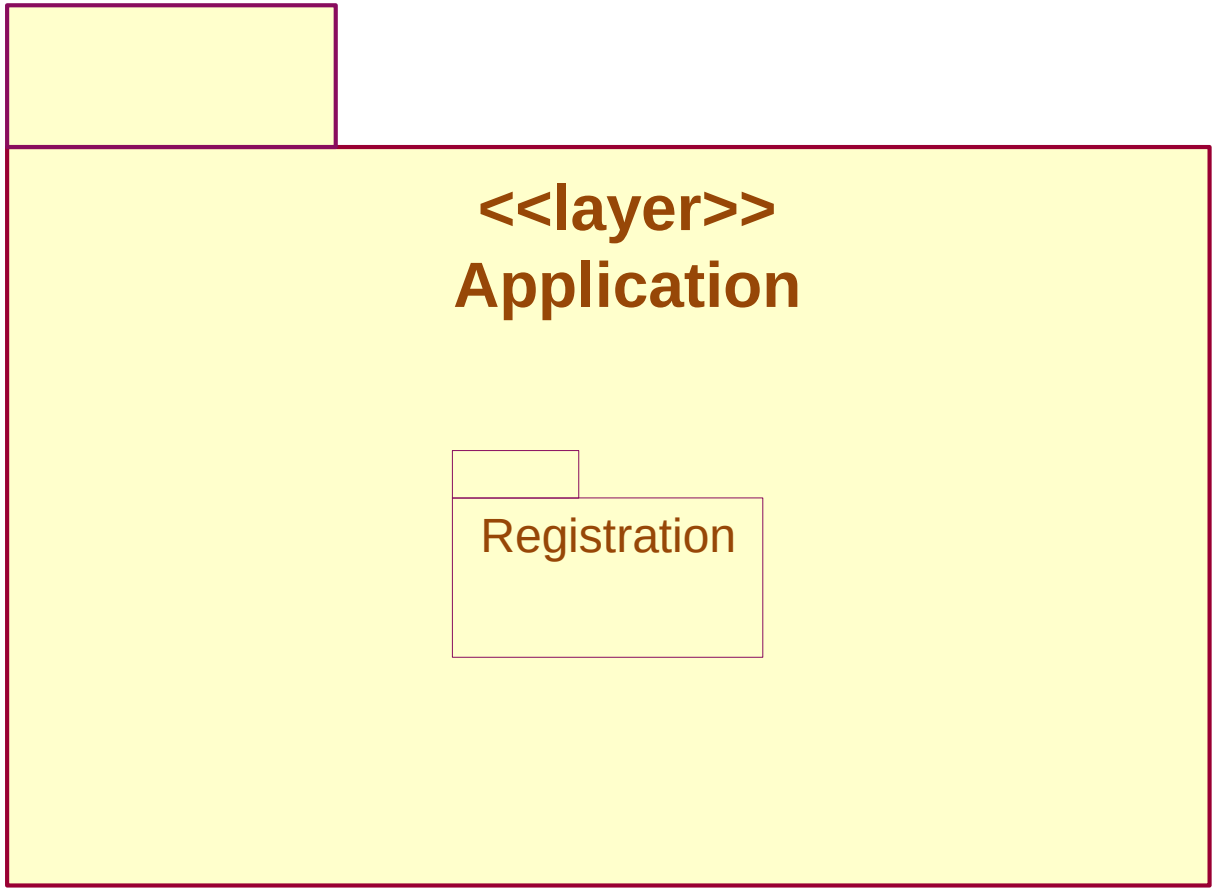
- Acoplamiento y cohesión
- Organización por usuario
- Áreas de competencia y / o de habilidad
- Distribución del sistema
- Secreto
- Variabilidad

Tratar de evitar dependencias ciclicas

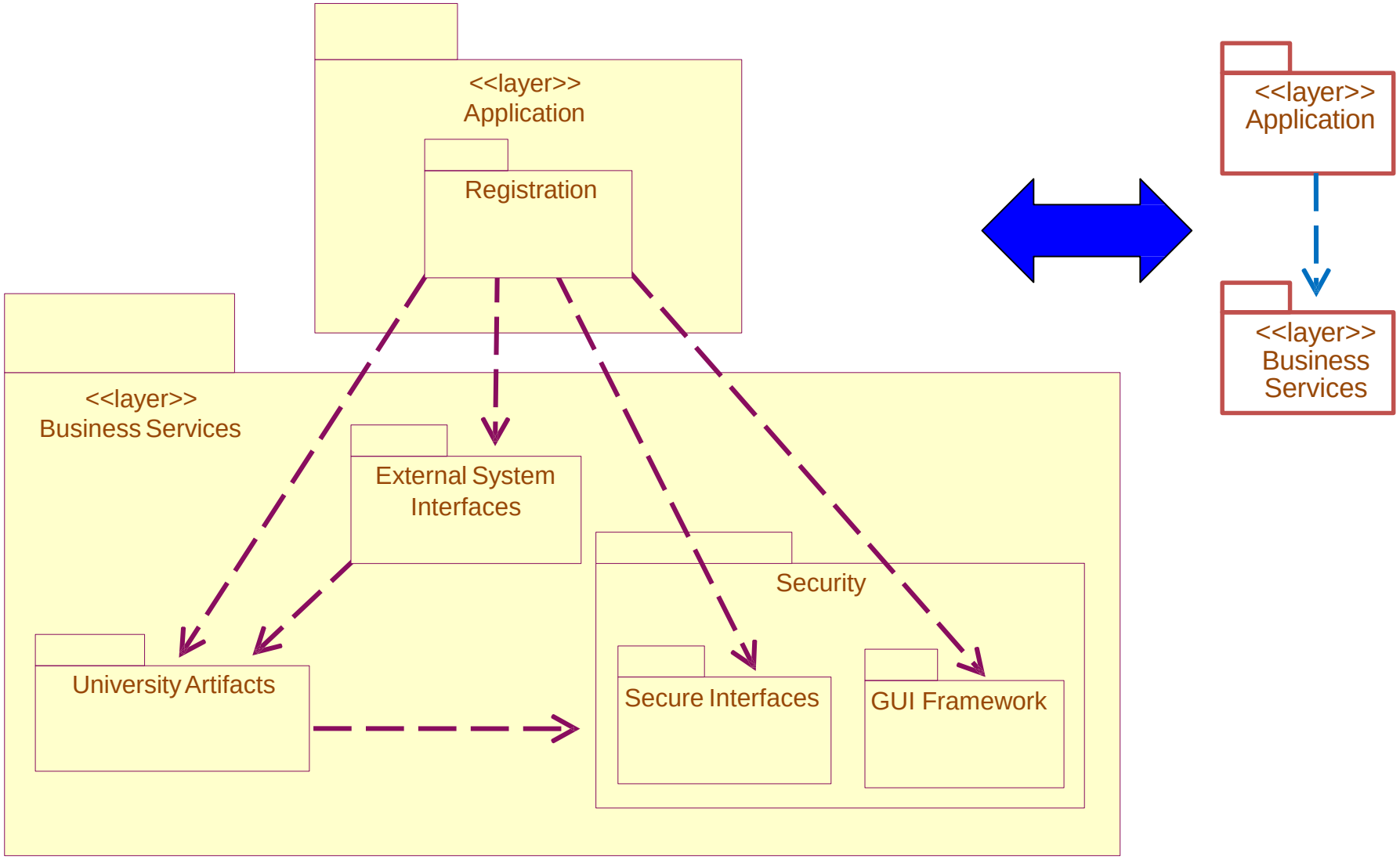
Ejemplo: Particionamiento



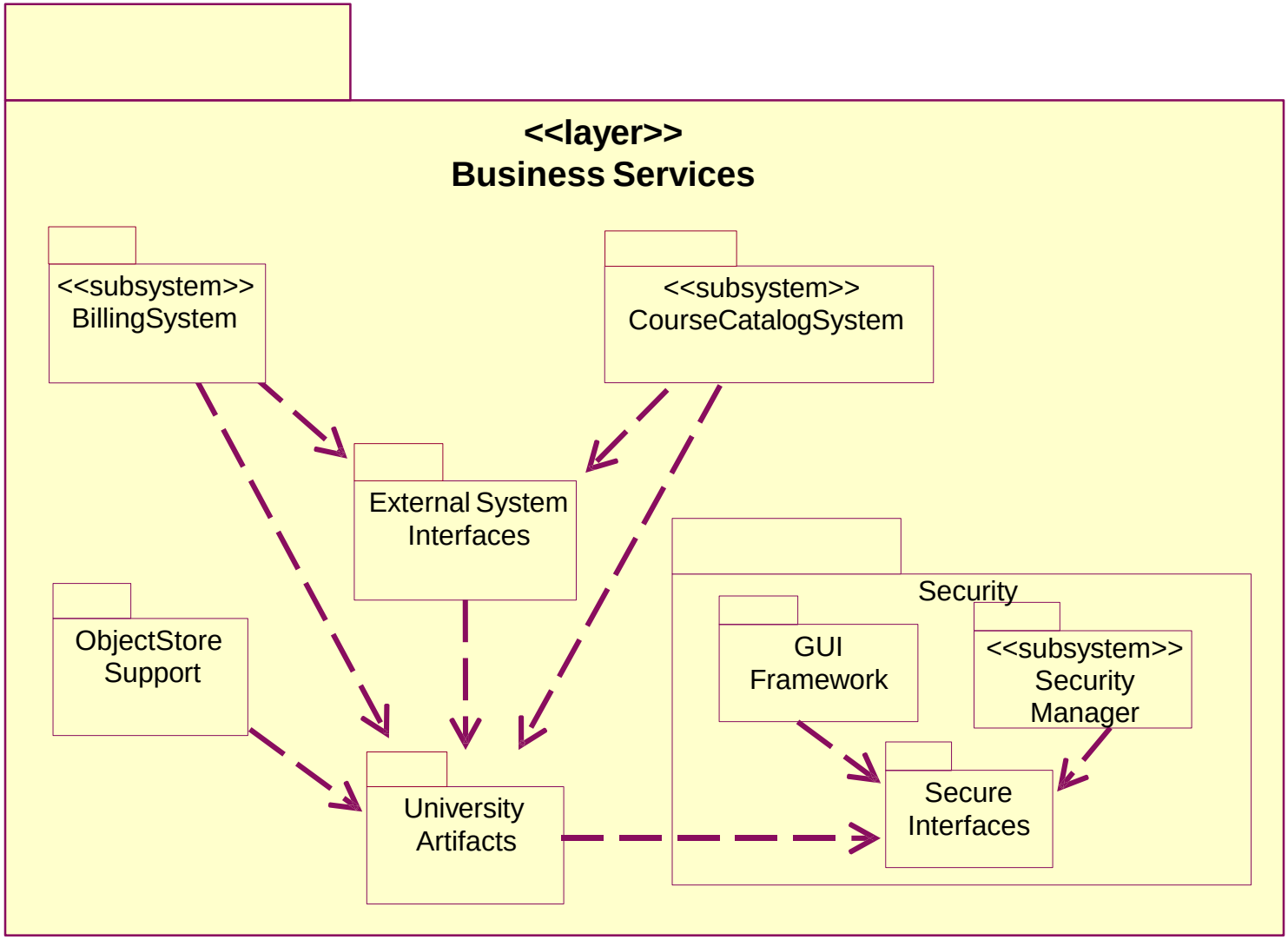
Ejemplo: Capa de aplicación



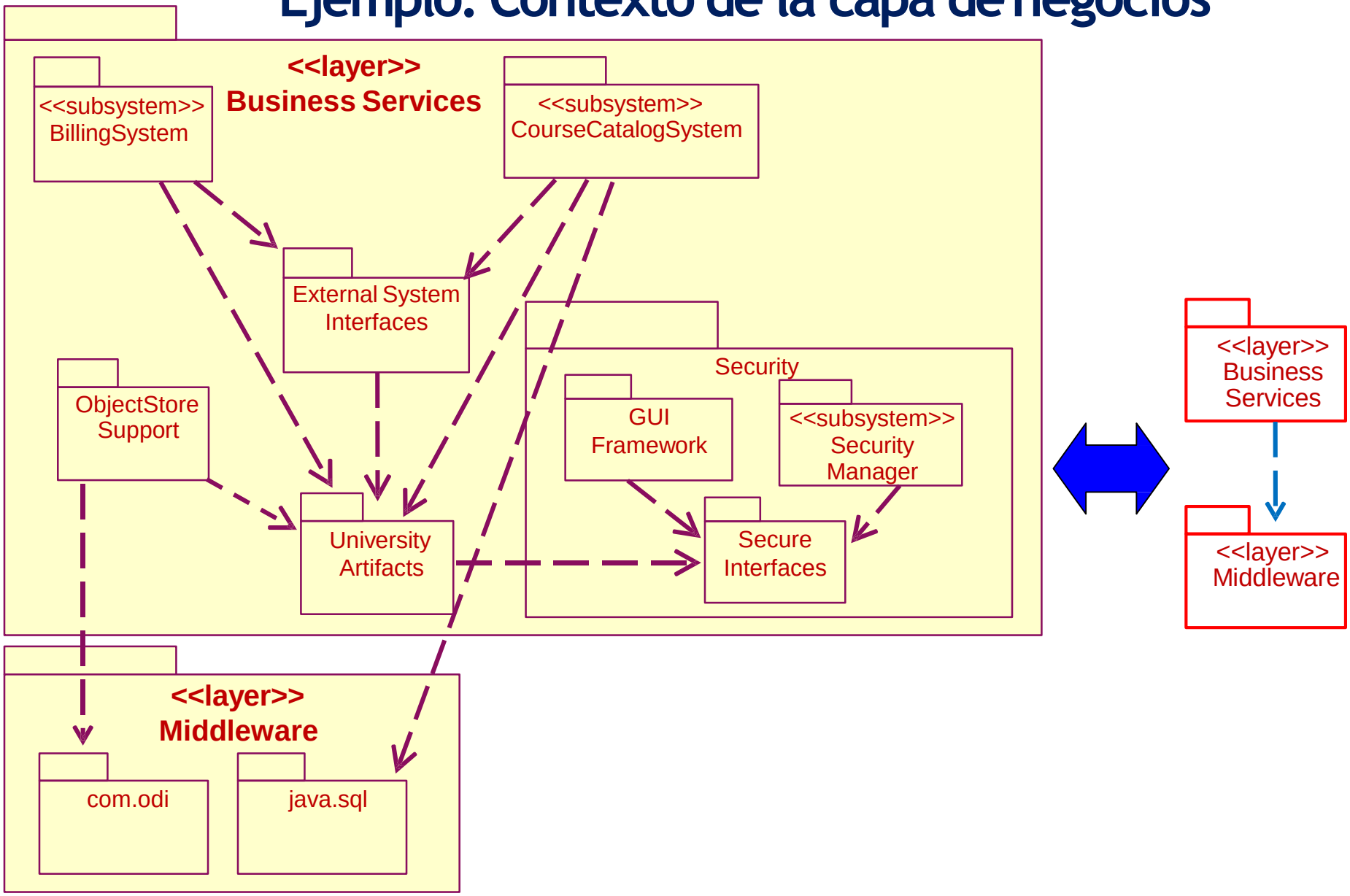
Ejemplo: Contexto de la capa de aplicación



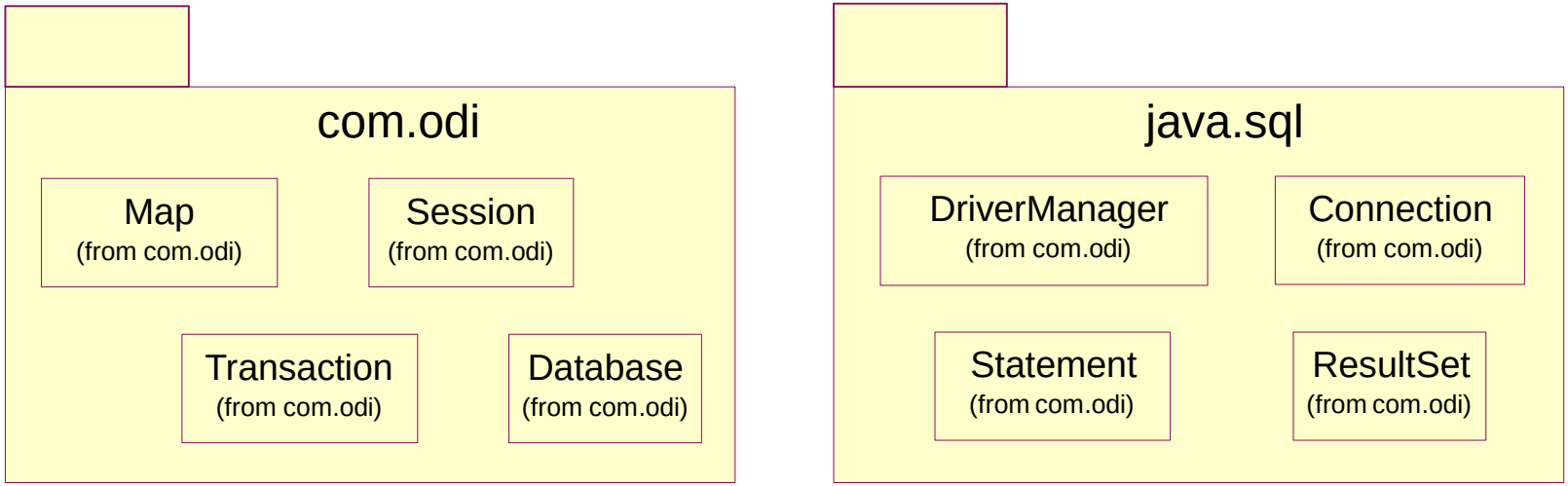
Ejemplo: Capa de servicios comerciales



Ejemplo: Contexto de la capa de negocios



Ejemplo de la capa Middleware



Checkpoints

- General
  - ¿Proporciona una visión global de los servicios de diferentes paquetes?
  - ¿Puedes encontrar las soluciones estructurales similares que pueden ser utilizados más ampliamente en el dominio del problema?
- Capas
  - ¿Hay más de siete capas?
- Subsistemas
  - ¿Se dividen los subsistemas de una manera lógica y consistente a través de todo el modelo?

Checkpoints (cont.)

- Paquetes
  - ¿Son los nombres de los paquetes descriptivos?
  - ¿Coincide la descripción del paquete con las responsabilidades de clases contenidas?
  - ¿Las dependencias de paquetes corresponden a las relaciones entre las clases contenidas?
  - ¿Las clases contenidas en un paquete corresponden a los criterios de la división del paquete?
  - ¿Existen clases o colaboraciones de clases dentro de un paquete que se puede separar en un paquete independiente?
  - Es la relación entre el número de paquetes y el número de clases apropiado?

Checkpoints (cont.)

- Classes
  - ¿El nombre de cada clase claramente refleja el papel que desempeña?
  - ¿Están las clases cohesionadas? es decir, ¿están todas las piezas acopladas funcionalmente?
  - ¿Son todos los elementos de la clase que necesitan las realizaciones de casos de uso?
  - ¿Los nombres de rol de las agrupaciones y asociaciones describen con precisión la relación?
  - ¿Son correctas las multiplicidades de las relaciones?

Repasemos: Elementos del diseño

- ¿Cuál es el propósito de identificar los elementos de diseño?
- ¿Qué es una interfaz?
- ¿Qué es un subsistema? ¿Cómo se diferencia de un paquete?
- ¿Qué es un subsistema utilizado para, y ¿cómo identificarlos?
- ¿Cuáles son algunas consideraciones de estratificación y partición?