# Diseño e implementación

1. Fundamentos del diseño de software
2. Diseño de la arquitectura
3. Diseño de los casos de uso
4. Diseño de la estructura de objetos

# ¿Qué es el diseño?

**Es donde se está con un pie en ambos mundos, el de la tecnología y el de las personas y los propósitos humanos, que tratan de unificarse**

**Vitruvio, crítico de arquitectura romano, afirmaba que los edificios bien diseñados eran aquellos que tenían resistencia, funcionalidad y belleza**

**Lo mismo se aplica al software**

**Resistencia:** Un programa no debe tener ningún error que impida su funcionamiento

**Funcionalidad:** Un programa debe ser apropiado para los fines que persigue

**Belleza:** La experiencia de usar un programa debe ser placentera

*Match Kapor, 1990*

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

- Definición y características
- Principios del diseño
- Herramientas de diseño
- Métodos de diseño
- Modelo del diseño
- Tareas del diseño

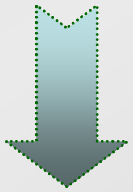
# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Definición y características

### Definición

El **diseño** es el proceso de aplicar distintas técnicas y principios con el propósito de definir un dispositivo, proceso o sistema con los suficientes detalles como para permitir su realización física

*Taylor, 1980*



El **diseño de software** es el proceso de aplicar distintas técnicas y principios de diseño con el propósito de traducir el modelo de análisis a una representación del software (modelo de diseño) que pueda codificarse

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

El **diseño** se puede definir como

(1) el proceso para definir la arquitectura, los componentes, las interfaces y otras características de un sistema o un componente

(2) el resultado de este proceso

*IEEE, 1990*

Parte funcional de un sistema que oculta su implementación proporcionando su realización a través de un conjunto de interfaces

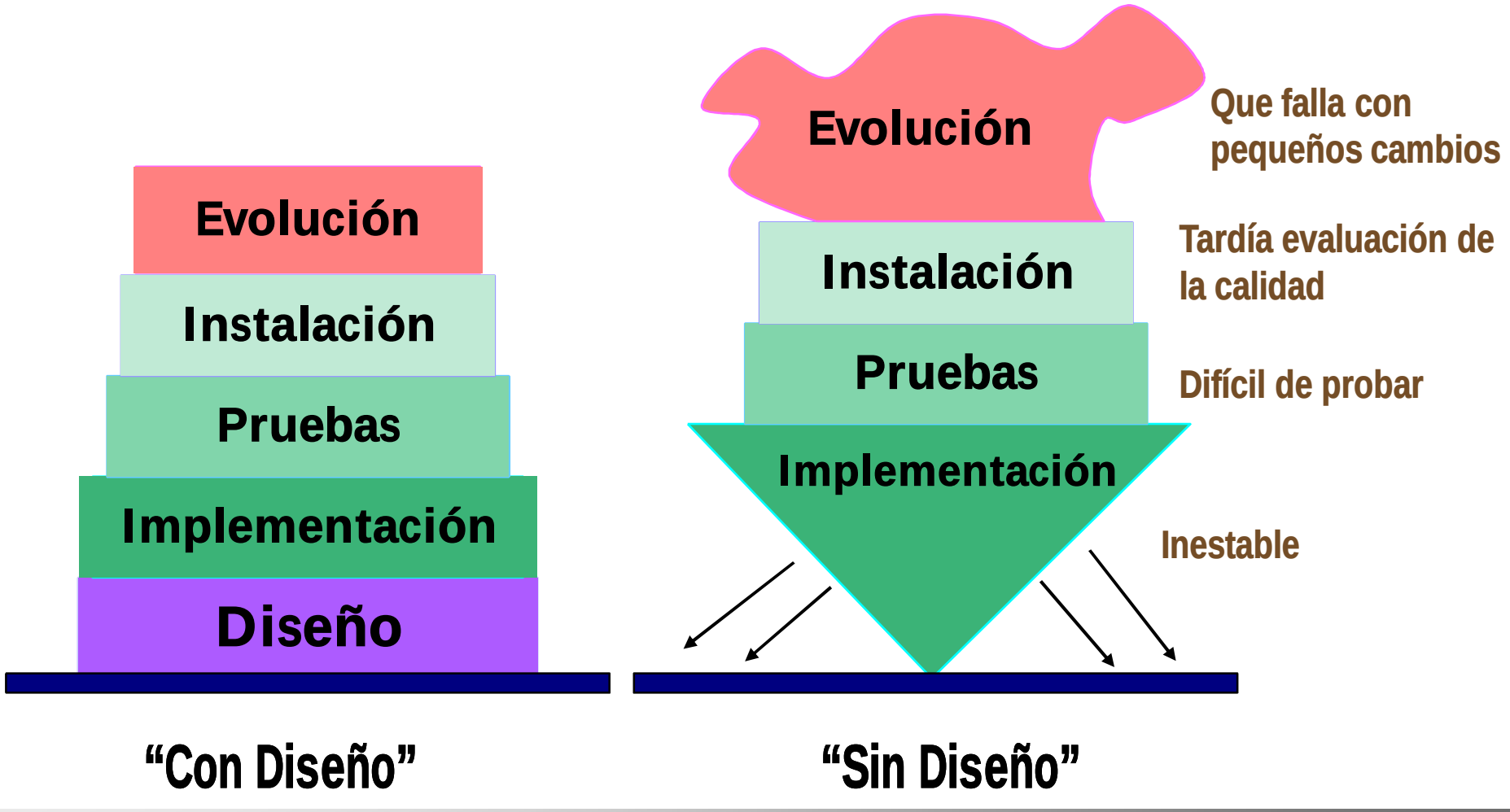
Describe la frontera de comunicación entre dos entidades software, definiendo explícitamente el modo en que un componente interacciona con otros

**El proceso de diseño se descompone en dos subprocesos**

- **Diseño arquitectónico**
  Se describe cómo descomponer el sistema y organizarlo en los diferentes componentes (arquitectura del software)
- **Diseño detallado**
  Se describe el comportamiento específico de cada uno de los componentes de software

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

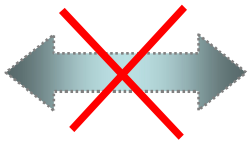
## Importancia del diseño



# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Características

- El diseño implica una propuesta de solución al problema especificado en el análisis
- Es una actividad creativa que se apoya en  la experiencia del diseñador
- Está apoyado por principios, técnicas, herramientas, ….
- Es un proceso clave para la calidad del producto software
- Es la base para el resto de etapas del desarrollo
- Es un proceso de refinamiento
- El diseño va a garantizar que un programa funcione correctamente

**Hacer que un programa funcione**  **Hacer que funcione CORRECTAMENTE**

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Principios del diseño

**Ayudan a responder a las siguientes preguntas**

- ¿Qué criterios se usan para dividir el software en sus componentes individuales?
- ¿Cómo se extraen los detalles de una función o estructura de datos a partir de la representación conceptual del software?
- ¿Cuáles son los criterios que definen la calidad técnica de un diseño software?

**Principios**

- **Abstracción**
- **División de problemas y modularidad**
- **Ocultamiento de información**
- **Independencia funcional**

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Abstracción

**Mecanismo que permite determinar qué es relevante y qué no lo es en un nivel de detalle determinado, ayudando a obtener la modularidad adecuada para ese nivel de detalle**

### Tipos de abstracciones

- **Abstracción de datos**
  Define un objeto compuesto por un conjunto de datos
  Ejemplo: La abstracción cliente, incluirá los datos de un cliente tal y como se entiende en la aplicación
- **Abstracción de control**
  Define un sistema de control sin describir información sobre su funcionamiento interno
  Ejemplo: Un semáforo para describir la coordinación en el funcionamiento de un sistema operativo
- **Abstracción procedimental**
  Se refiere a la secuencia de pasos que conforman un proceso determinado
  Ejemplo: Un algoritmo de ordenación

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## División de problemas y modularidad

La división de problemas sugiere que cualquier problema complejo puede manejarse con más facilidad si se subdivide en elementos susceptibles de resolverse u optimizarse de manera independiente

**Matemáticamente, esto se explica:**

$C(x)$ función que define la complejidad de un problema x

$E(x)$ función que define el esfuerzo de desarrollo de un problema x

Si para dos problemas $p_1$ y $p_2$ $C(p_1) > C(p_2)$ se deduce que $E(p_1) > E(p_2)$

Además se cumple $C(p_1 + p_2) > C(p_1) + C(p_2)$ y que $E(p_1 + p_2) > E(p_1) + E(p_2)$

**DIVIDE Y VENERÁS**

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

**La modularidad es la manifestación más común de la división de problemas**

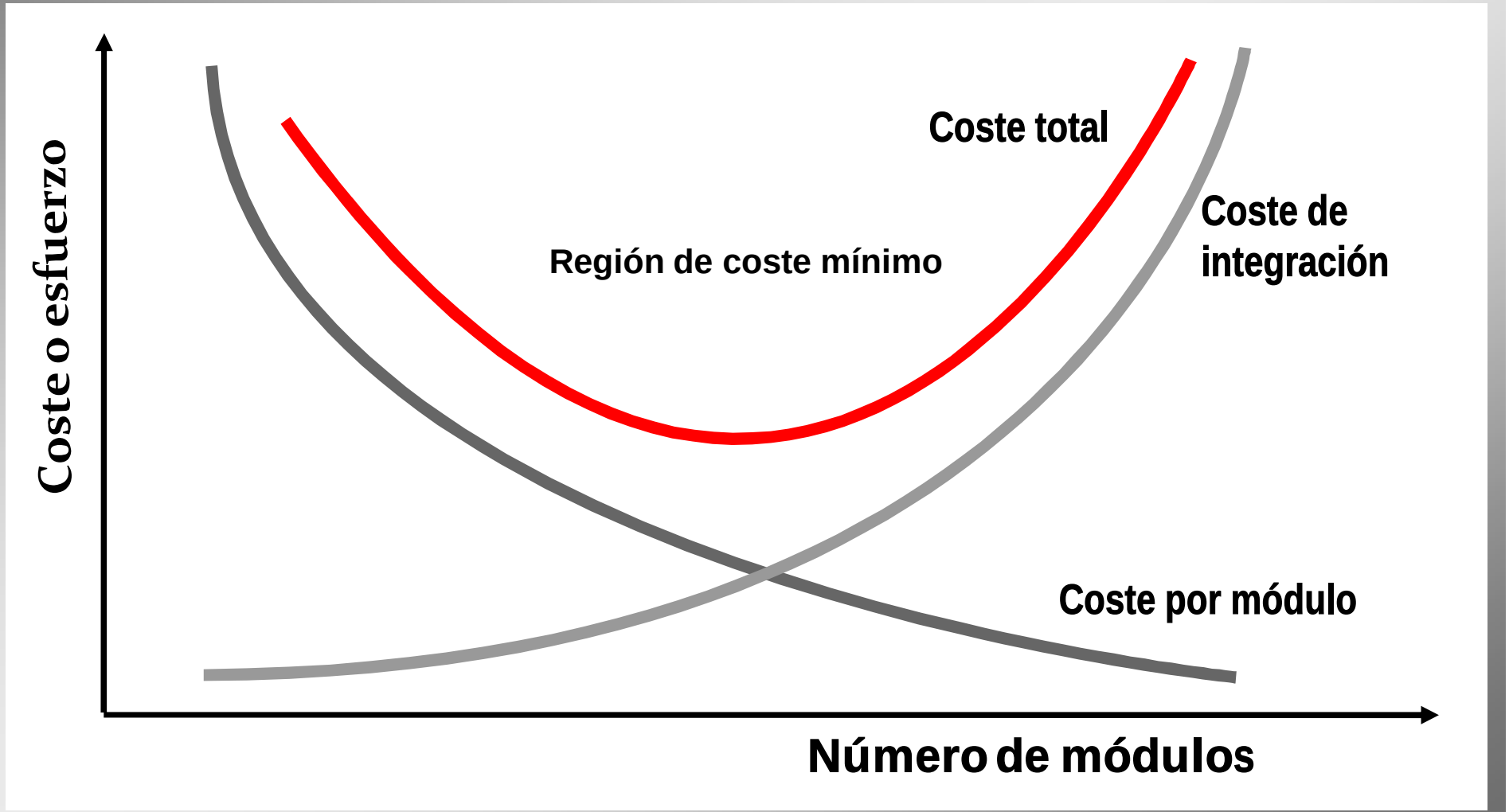
**El software se divide en componentes con nombres distintos y abordables por separado, denominados módulos, que se integran para satisfacer los requisitos del problema**

**Ventajas de la modularidad**

- Son más fáciles de entender y de documentar que todo el subsistema o sistema
- Facilitan los cambios
- Reducen la complejidad
- Proporcionan implementaciones más sencillas
- Posibilitan el desarrollo en paralelo
- Permiten la prueba independiente (prueba de unidad)
- Facilitan el encapsulamiento

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

**Grado adecuado de modularidad**



# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Ocultamiento de información

**¿Cómo descomponer una solución software para obtener el mejor conjunto de módulos?**

> **Los módulos deben especificarse y diseñarse, de forma que la información (algoritmos y datos) contenida en un módulo sea inaccesible para los que no necesiten de ella**

### Ventajas del ocultamiento de información

- Reduce la probabilidad de “efectos colaterales”
- Limita el impacto global de las decisiones de diseño locales
- Enfatiza la comunicación a través de interfaces controladas
- Disminuye el uso de datos globales
- Potencia la modularidad
- Produce software de alta calidad

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Independencia funcional

El software debe diseñarse de manera que cada módulo resuelva un subconjunto específico de requisitos y tenga una interfaz sencilla cuando se vea desde otras partes de la estructura del programa

**La independencia se evalúa con dos criterios**

- **Cohesión**

  Un módulo cohesivo ejecuta una sola tarea, por lo que requiere interactuar poco con otros componentes en otras partes del programa. Idealmente hace una sola cosa

  La ALTA COHESIÓN proporciona módulos fáciles de entender, reutilizar y mantener

- **Acoplamiento**

  Indica la interconexión entre módulos en una estructura de software y depende, básicamente, de la complejidad de la interfaz entre módulos

  El BAJO ACOPLAMIENTO proporciona módulos fáciles de entender y con menos efectos colaterales

Máxima cohesión y mínimo acoplamiento

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Herramientas de diseño

Instrumentos que ayudan a representar los modelos de diseño de software

Algunas de las más usuales

- Diagramas de UML: de clase, de interacción, de paquetes, de despliegue, ...
- Cartas de estructura (o diagramas de estructura)
- Tablas de decisión (o tablas de transiciones)
- Diagramas de flujo de control (u organigramas estructurados)
- Diagramas de Nassi-Shneiderman (o diagramas NS o de Chaplin)
- Lenguajes de diseño de programas (LDP o pseudocódigo)

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Métodos de diseño

**Proporcionan las herramientas, técnicas y los pasos a seguir para obtener diseños de forma sistemática**

### Características

- Principios en los que se basa
- Mecanismos de traducción del modelo de análisis al modelo de diseño
- Herramientas que permiten representar los componentes funcionales y estructurales
- Heurísticas que permiten refinar el diseño
- Criterios para evaluar la calidad del diseño

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

**Principales métodos de diseño**

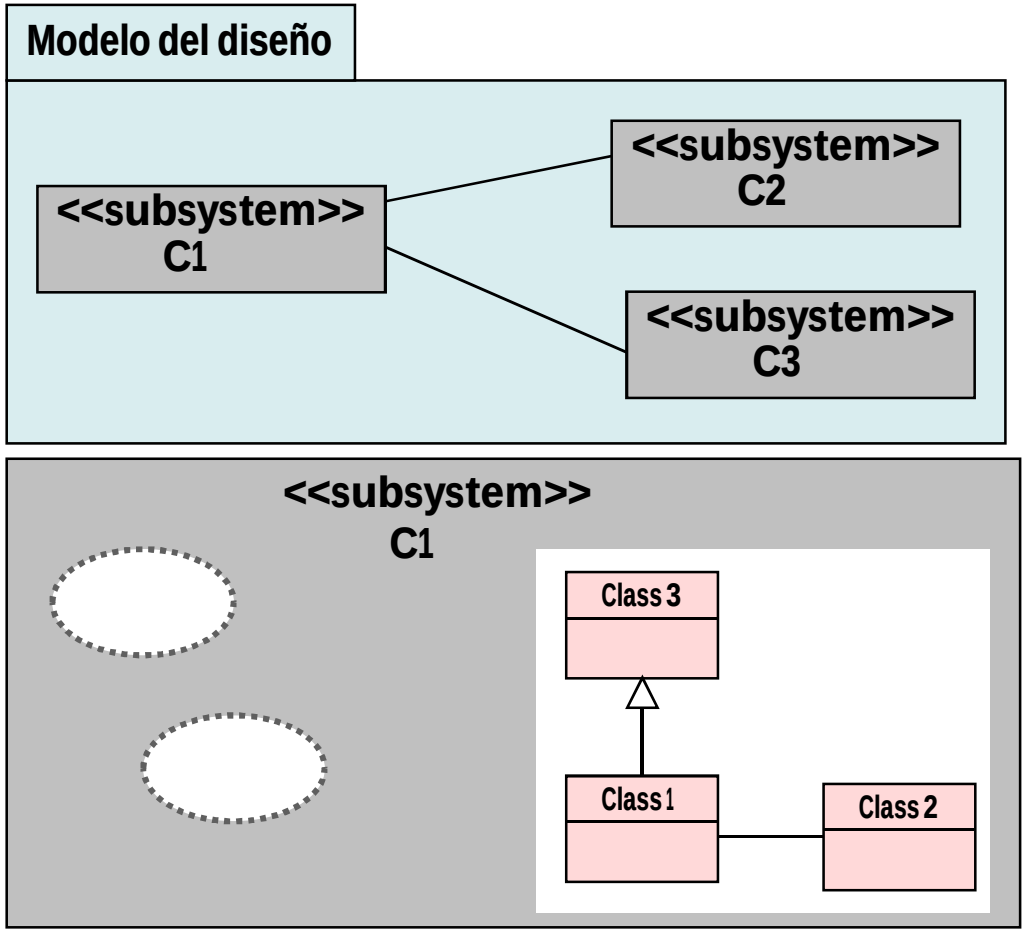
- **SSD (Diseño Estructurado de Sistemas)**
- **JSD (Desarrollo de Sistemas Jackson)**
- **ERA (Entidad-Relación-Atributo)**
- **OMT (Técnicas de Modelado de Objetos)**
- **Método Booch (Método de diseño orientado a objetos)**
- **Métodos orientados a objetos**

  La gran mayoría usan cono herramienta de modelado UML y como proceso de desarrollo el PU

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Modelo del diseño

### Contenido del modelo




A nivel general está formado por varios subsistemas de diseño junto con las interfaces que requieren o proporcionan estos subsistemas

Cada subsistema de diseño puede contener diferentes tipos de elementos de modelado de diseño, principalmente realización de casos de uso-diseño y clases de diseño

# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Relación con el modelo de análisis

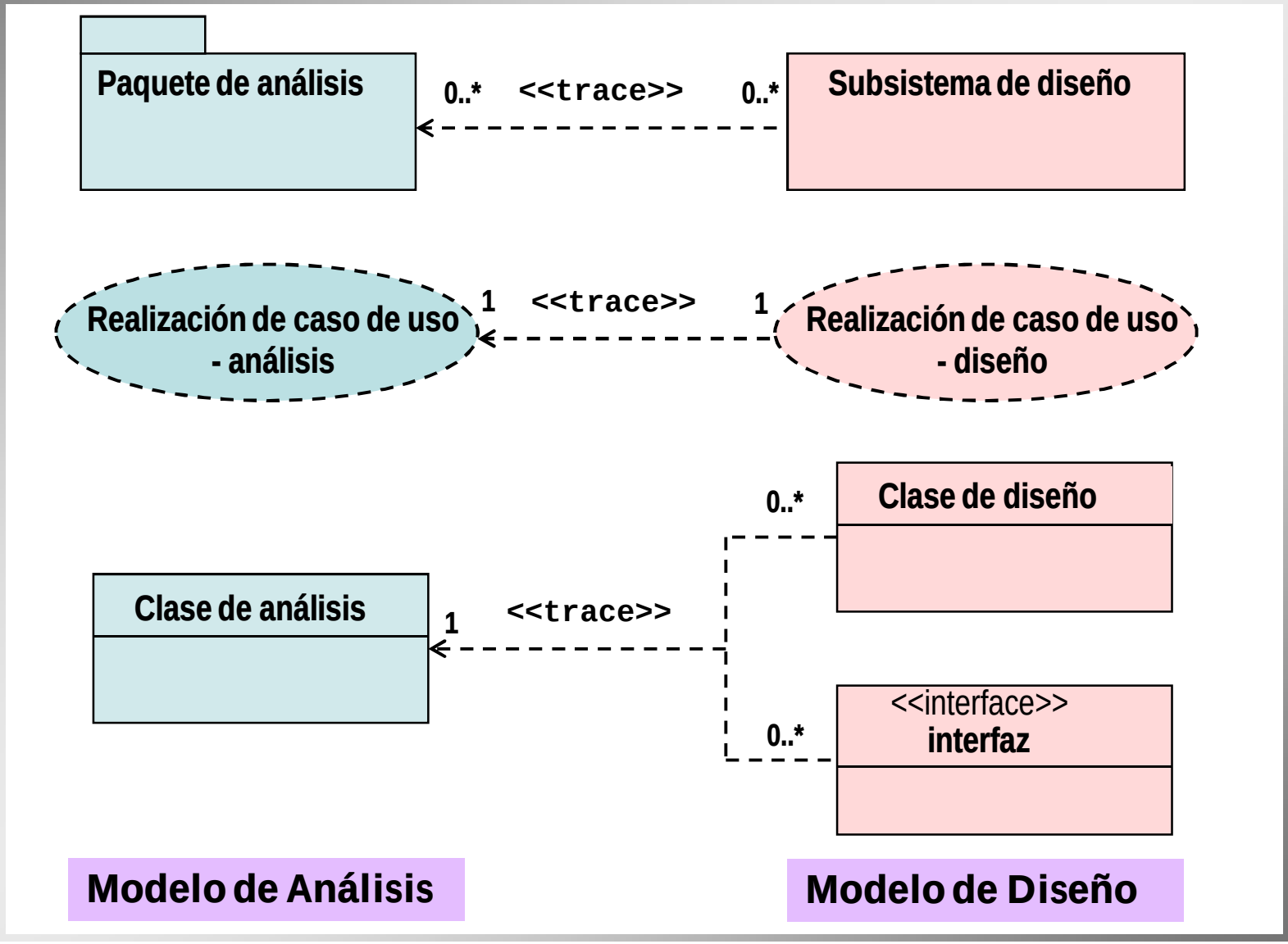
Se puede pensar en el modelo de diseño como una elaboración del modelo de análisis con detalles añadidos y soluciones técnicas específicas


¿Qué hacer con estos dos modelos?

| Estrategia | Consecuencias |
|---|---|
| (1) Convertir el modelo de análisis en un modelo de diseño | Se tiene un modelo de diseño pero se pierde la vista de análisis |
| (2) Convertir el modelo de análisis en un modelo de diseño y usar una herramienta para recuperar una "vista de análisis" | Se tiene un modelo de diseño pero la vista recuperada del modelo de análisis puede no ser satisfactoria |
| (3) Congelar el modelo de análisis y hacer una copia en un modelo de diseño | Se tienen dos modelos pero no van al mismo ritmo |
| (4) Mantener los dos modelos separados | Se tienen dos modelos, van al mismo ritmo, pero existe una carga de mantenimiento |

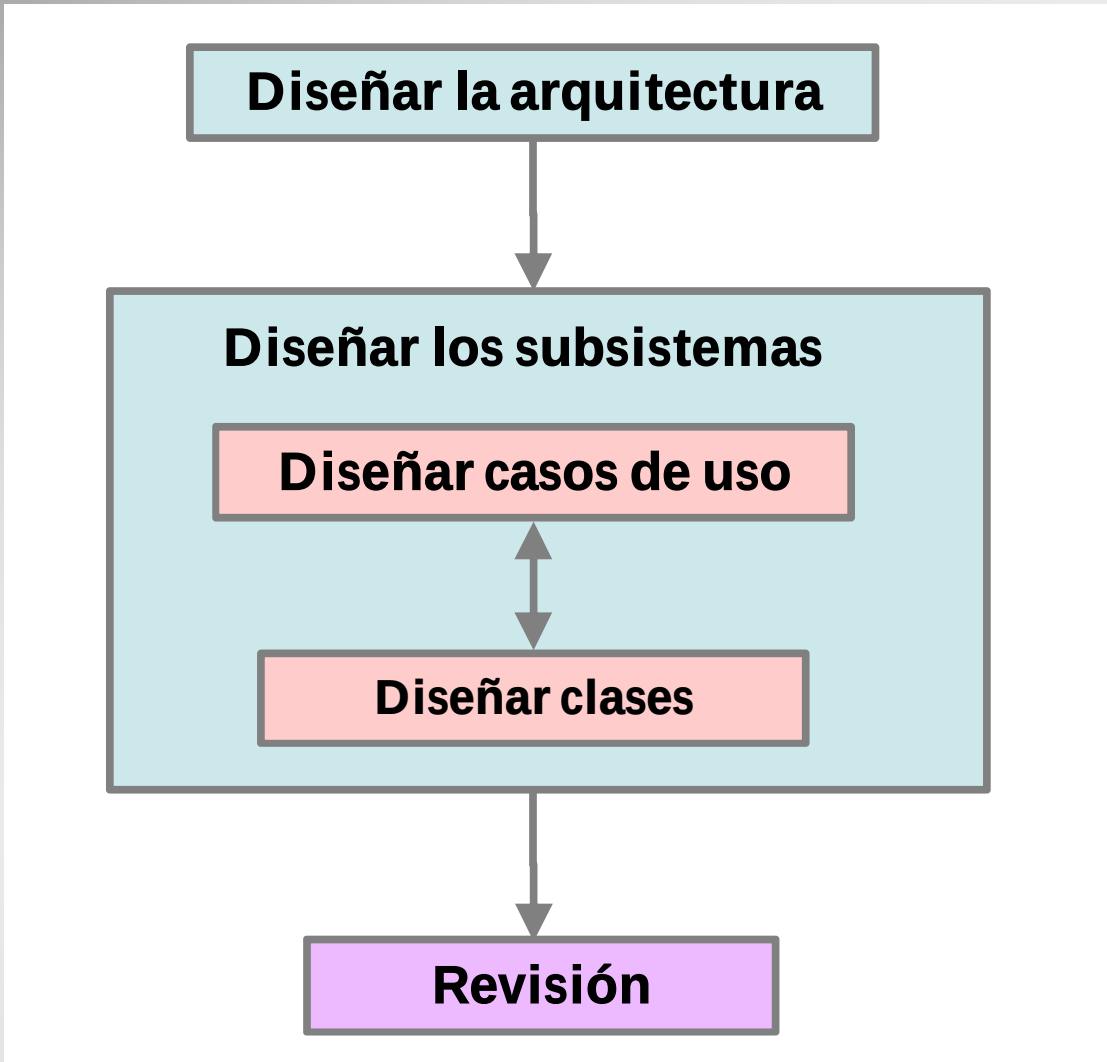
# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

## Trazabilidad entre los distintos elementos del análisis y del diseño

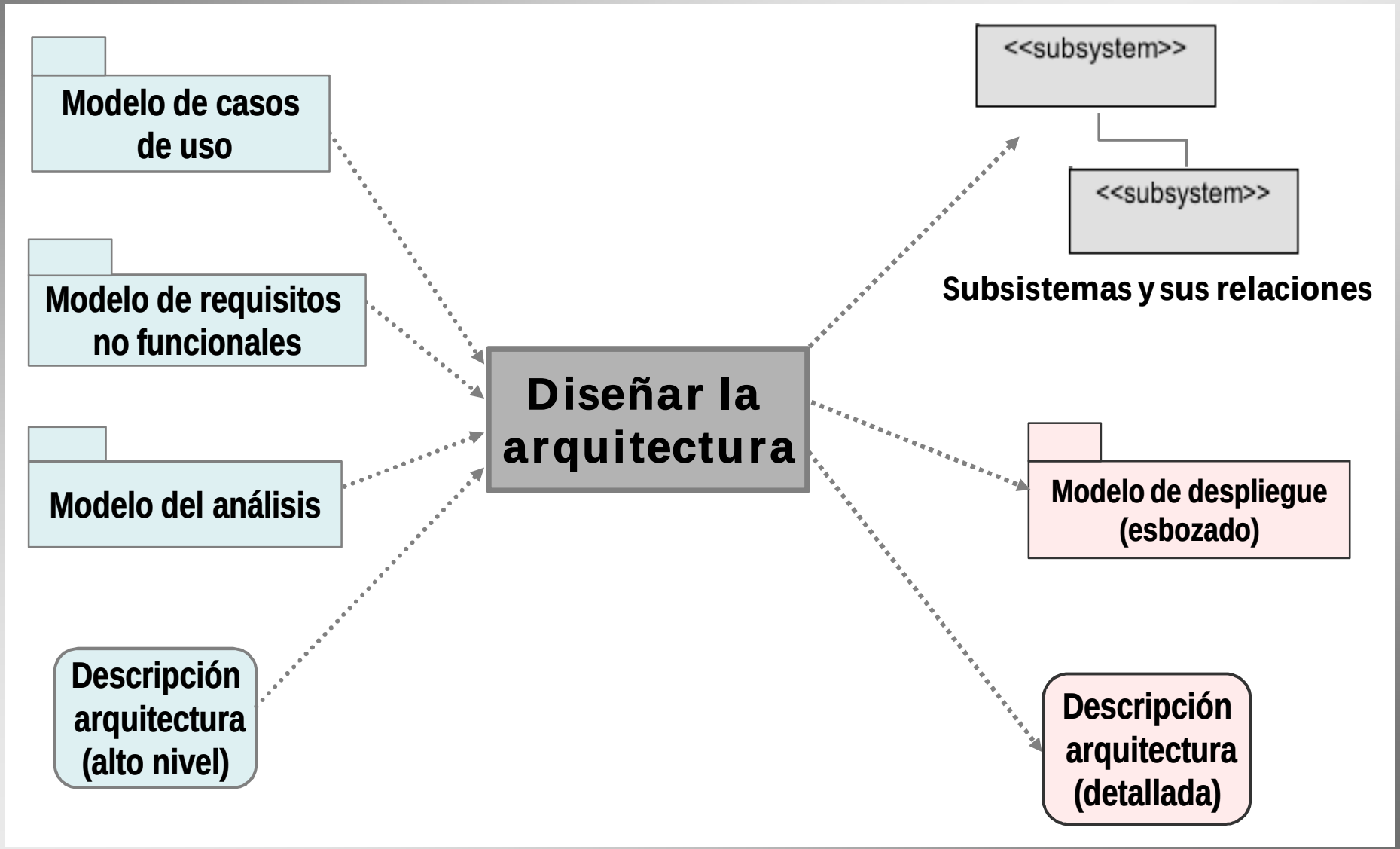


# FUNDAMENTOS DEL DISEÑO DE SOFTWARE

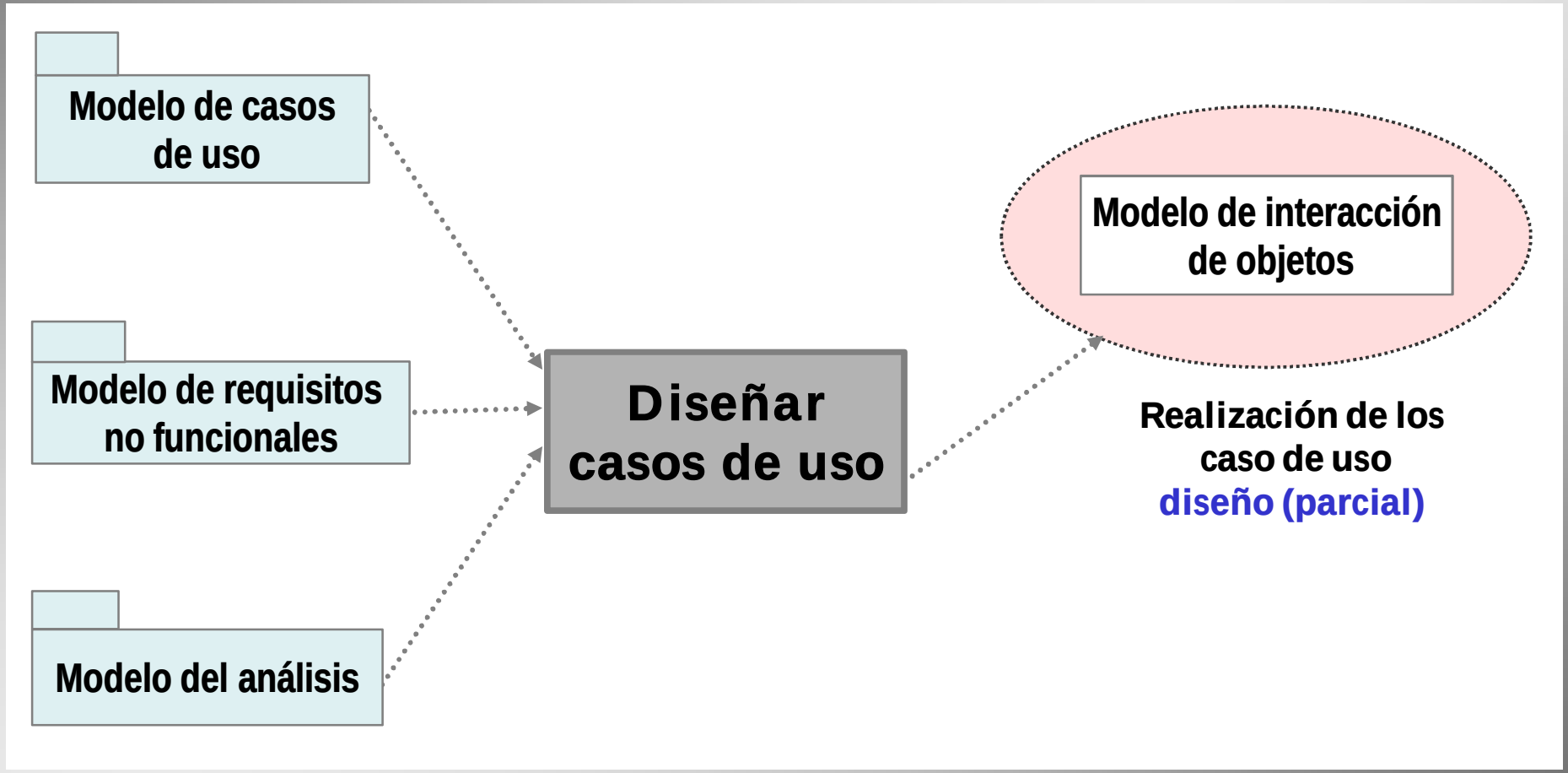
## Tareas del diseño



# FUNDAMENTOS DEL DISEÑO DE SOFTWARE



# FUNDAMENTOS DEL DISEÑO DE SOFTWARE



# FUNDAMENTOS DEL DISEÑO DE SOFTWARE