Diseño e implementación

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

Fundamentos del diseño software

Contenido

Definición y características

Principios del diseño

Herramientas del diseño

Métodos de diseño

Modelo del diseño

Tareas del diseño

¿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 errores que impida su funcionamiento

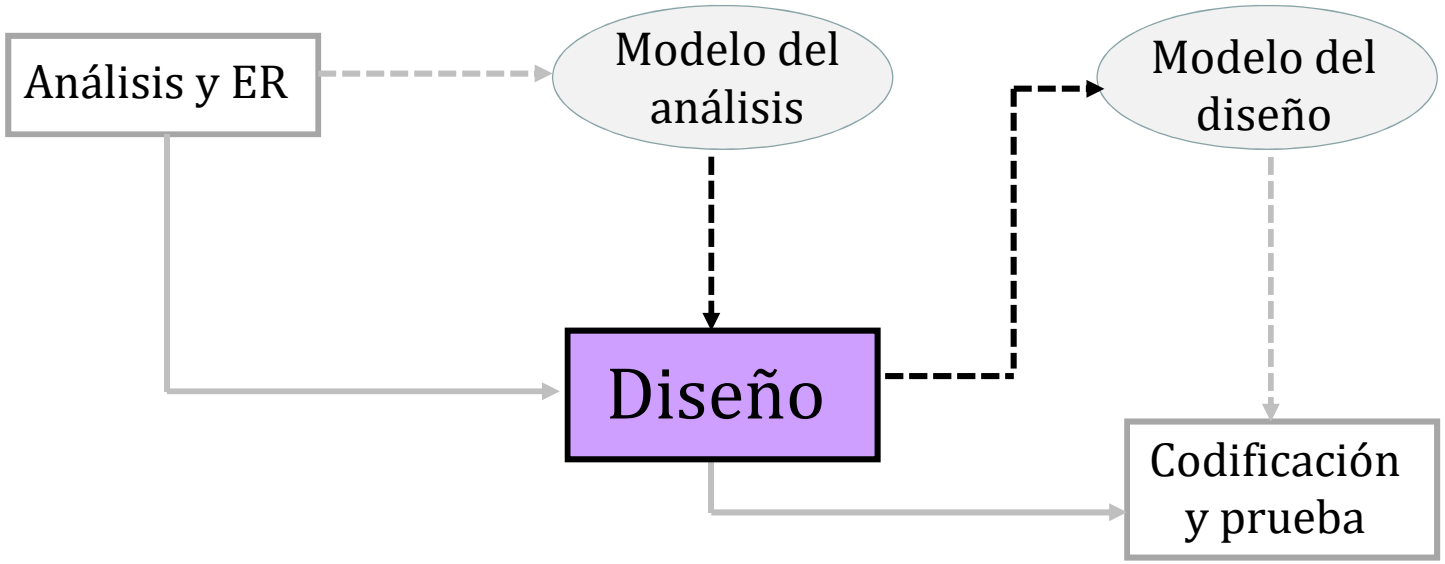
Funcionalidad: Un programa debe ser apropiado para los fines que persigue

Belleza: La experiencia de usar un programa debe ser placentera

Match Kapor, 1990

Definición y características

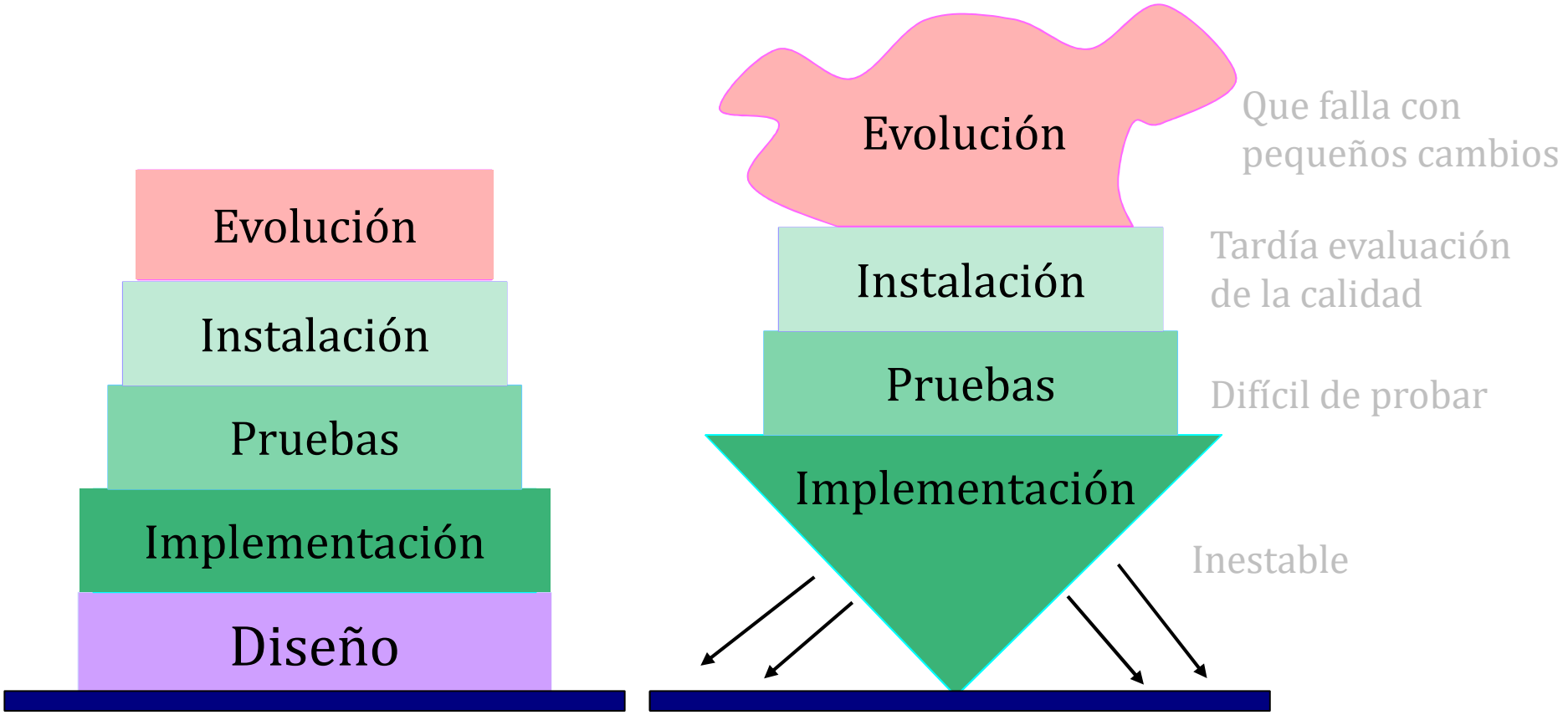
El diseño es el proceso de aplicar distintas métodos, herramientas y principios con el propósito de definir un dispositivo, proceso o sistema con los suficientes detalles como para permitir su realización física



El diseño de software es el proceso de aplicar métodos, herramientas y principios de diseño para traducir el modelo de análisis a una representación del software (modelo de diseño) que pueda codificarse

Definición y características

El diseño es fundamental

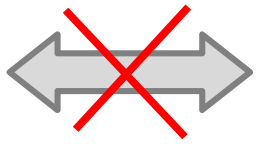


"Con Diseño"

"Sin Diseño"

Definición y 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 una tarea clave para la calidad del producto software
- Es la base para el resto de etapas del desarrollo
- Debe ser un proceso de refinamiento
- El diseño va a garantizar que un programa funcione correctamente

Hacer que un programa funcione  Hacer que funcione CORRECTAMENTE

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

- División de problemas y modularidad
- Abstracción
- Ocultamiento de información
- Independencia modular

Principios del diseño

División de problemas y modularidad

Divide y vencerás

"Un sistema software debe estar formado por piezas (Módulos), que deben encajar perfectamente, que interactúan entre sí para llevar a cabo algún objetivo común"

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 $E(p_1 + p_2) > E(p_1) + E(p_2)$

Principios del diseño

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

Un Módulo Software es una unidad básica de descomposición de un sistema software y representa una entidad o un funcionamiento específico

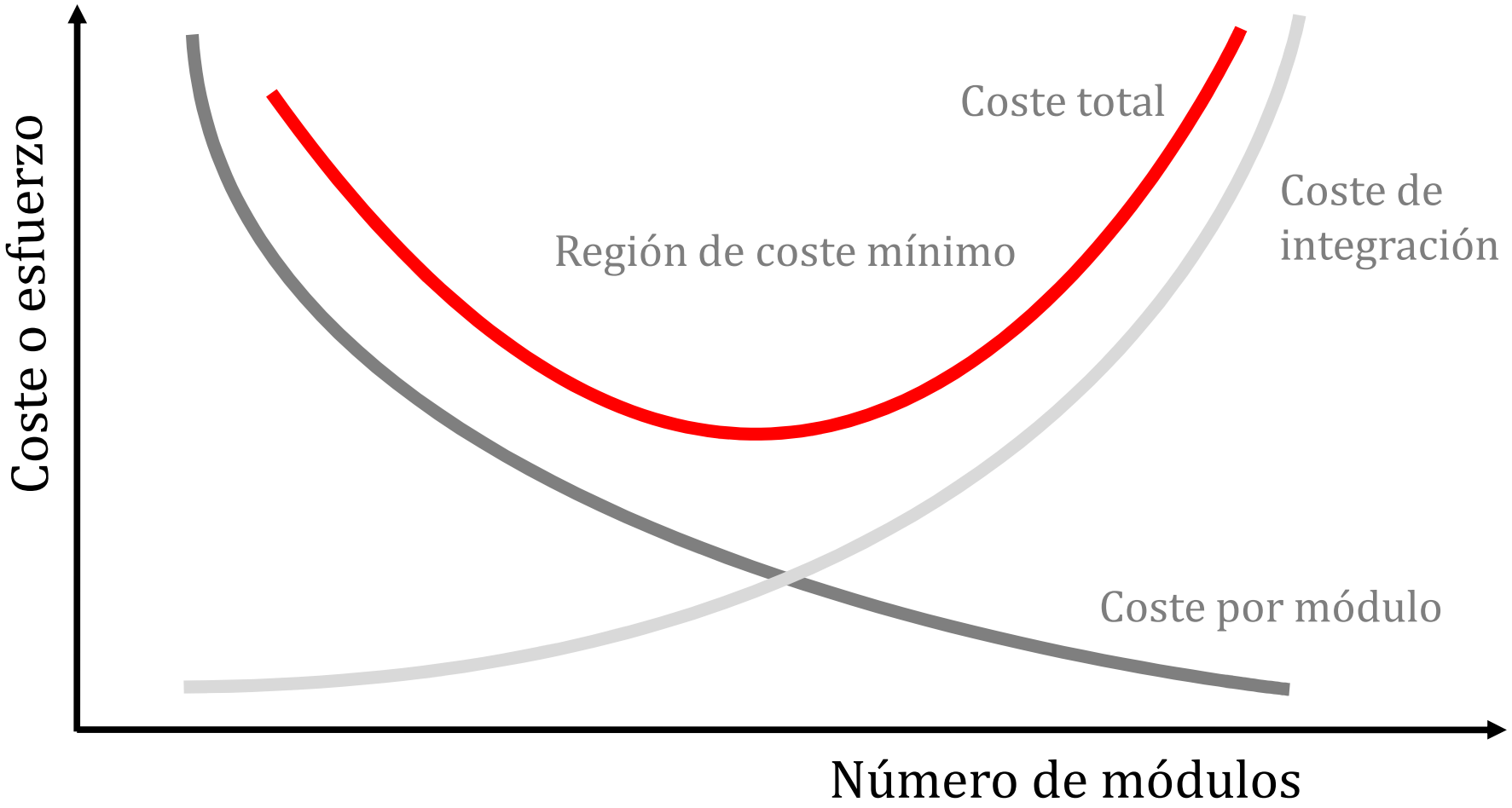
Pueden ser Módulos: una función, una clase, un paquete, …

Ventajas de la modularidad

- Son más fáciles de entender y documentar que todo el subsistema
- 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

Principios del diseño

Grado adecuado de modularidad



Principios del diseño

Abstracción

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

Mecanismos de abstracción en el diseño

- Abstracción procedimental
- Abstracción de datos
- Abstracción de control

Al proceso de ir incorporando detalles al diseño conforme se vaya bajando el nivel de abstracción se denomina REFINAMIENTO

(N. Wirth, 1971)

Principios del diseño

Abstracción procedimental

Se abstrae sobre el funcionamiento para conseguir una estructura modular basada en procedimientos

Se refiere a la secuencia de pasos que conforman un proceso determinado

Ejemplo: Un algoritmo de ordenación

Abstracción de datos

Se abstrae tanto el funcionamiento como los atributos que definen el estado de una entidad, para obtener una estructura modular basada en el estado y funcionamiento de una entidad u objeto

Define un objeto (entidad) compuesto por un conjunto de datos

Ejemplo: La entidad cliente incluirá los datos de un cliente tal y como se entiende en la aplicación

Abstracción de control

Se abstrae sobre el flujo de control de cualquier proceso en general

Define un sistema de control sin describir su funcionamiento interno

Ejemplo: Un semáforo para describir la coordinación en el funcionamiento de un S.O.

Principios del diseño

Ocultamiento de información

"Un módulo debe especificarse y diseñarse de forma que la información (procedimientos y datos) contenida en el módulo sea inaccesible para otros módulos que no necesiten esa información"

Beneficios 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

Principios del diseño

Independencia modular

"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 de un módulo se mide con dos parámetros: cohesión y acoplamiento

Cohesión

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

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

Si el módulo es una clase, ésta presenta un nivel alto de cohesión si modela un solo concepto abstracto con un pequeño conjunto de responsabilidades íntimamente relacionadas y todas sus operaciones, atributos y asociaciones están para realizarlas

Principios del diseño

Acoplamiento

Medida de la interdependencia entre módulos dentro de una estructura de software. Un módulo debe presentar un nivel de acoplamiento, con los demás módulos lo más bajo posible

Un grado adecuado de acoplamiento entre módulos es indispensable, y hay que:

- Tratar de reducirlo siempre que sea posible
- Mostrarlo de forma explícita en todos los modelos del diseño

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

Si el módulo es una clase, ésta debería relacionarse (mediante herencia, asociación o dependencia) solo con las clases que necesite para llevar a cabo sus responsabilidades

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 componentes, de despliegue, …
- Cartas de estructura
- Tablas de decisión
- Diagramas de flujo de control

  Organigramas estructurados, Diagramas de NS (Nassi-Shneiderman)
- Lenguajes de diseño de programas (LDP)

Métodos de diseño

Permiten obtener diseños de forma sistemática, proporcionando las herramientas, las técnicas y los pasos a seguir para llevar a cabo el diseño

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

Métodos de diseño

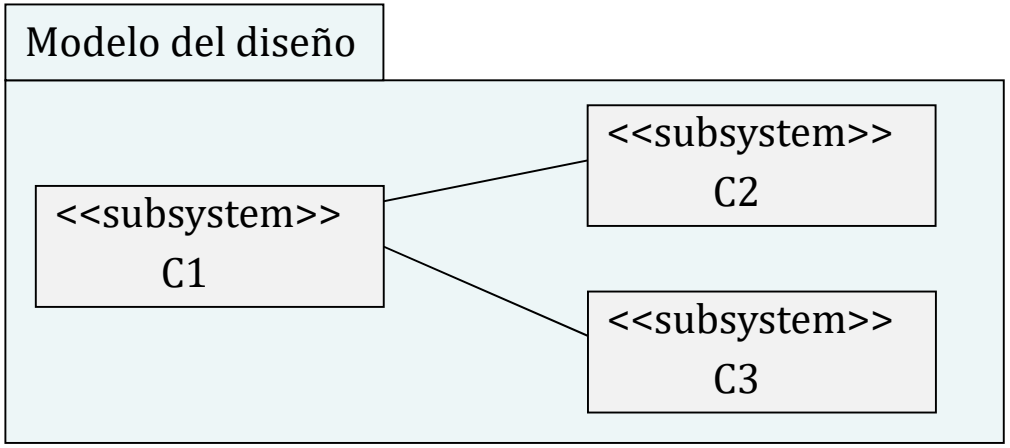
Principales métodos de diseño

- SSD (Diseño estructurado de sistemas)
- JSD (Desarrollo de sistemas Jackson)
- ERA (Entidad-Relación-Atributo)
- OMT (Técnicas de modelado de objetos)
- Método de Booch (Método de diseño basado en objetos)
- Métodos orientados a objetos

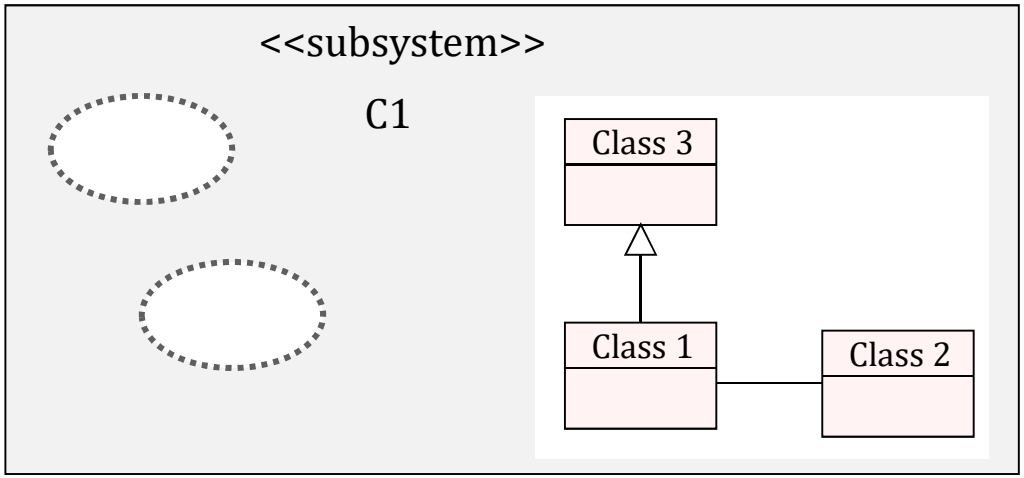
  En la actualidad existe una gran variedad de métodos orientados a objetos, aunque la mayoría de ellos usan como herramienta de modelado UML y como proceso de desarrollo el PU

Modelo de diseño

Contenido del modelo



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

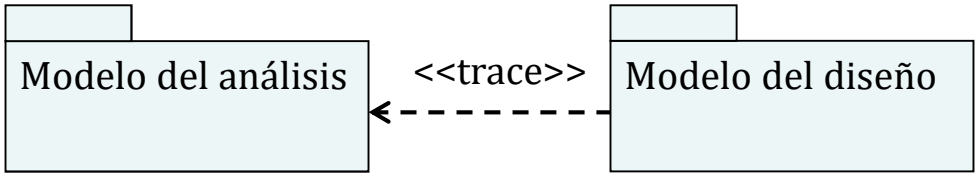


Cada subsistema de diseño puede contener diferentes tipos de elementos de modelado de diseño, principalmente realización de casos de uso-diseño y clases de diseño

Modelo de diseño

Relación con el modelo de análisis

El modelo de diseño se puede considerar como una elaboración o refinamiento del modelo de análisis, en el que todos los artefactos están mejor definidos e incorporan detalles técnicos que permiten su implementación

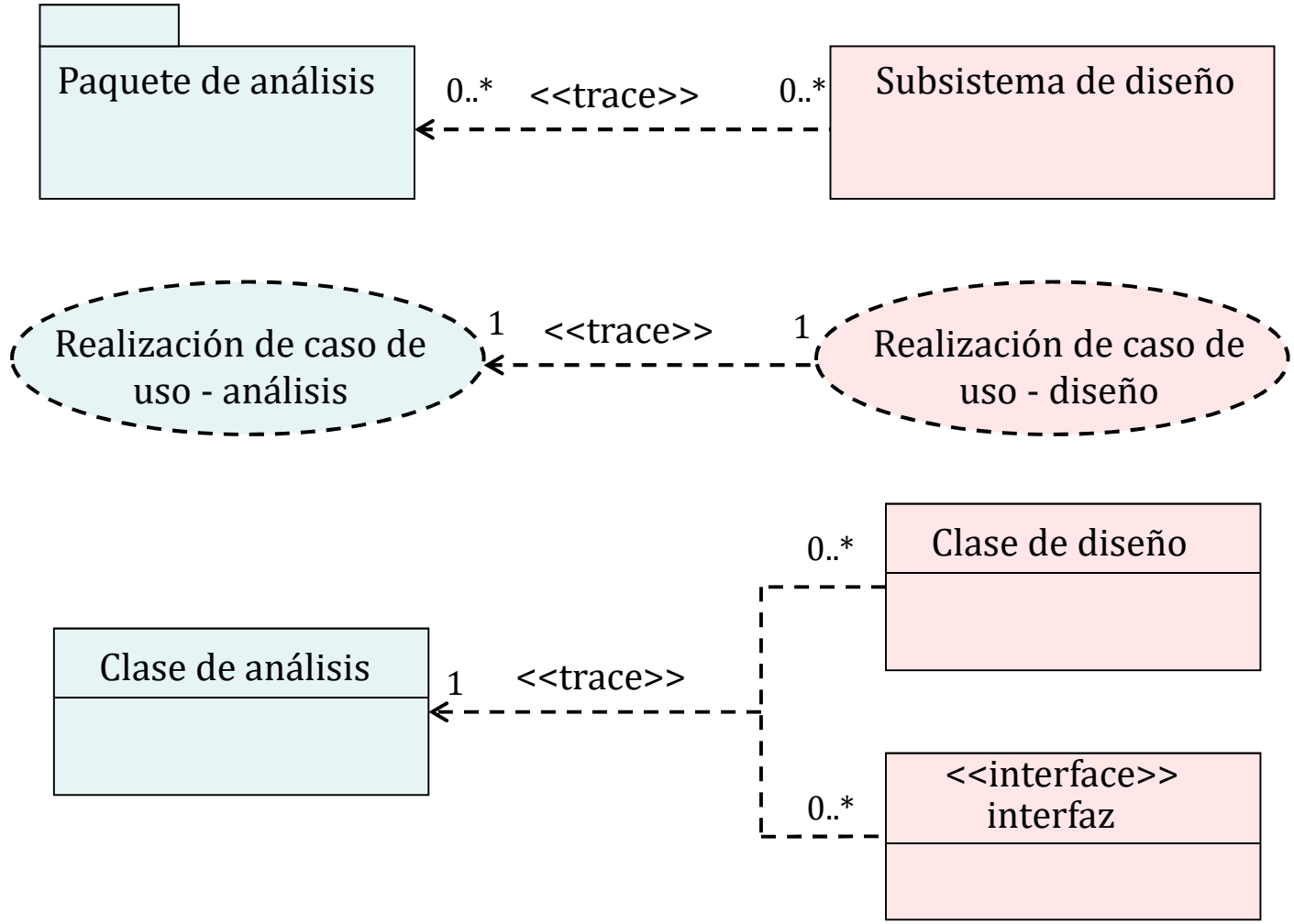


¿Cómo conseguir la trazabilidad entre modelos?

Estrategia	Consecuencias
A) Convertir el modelo de análisis en un modelo de diseño	Se tiene un modelo de diseño pero se pierde la vista de análisis
B) Igual que A y usar una herramienta para recuperar el modelo de análisis	Se tiene un modelo de diseño pero la vista recuperada del modelo de análisis puede no ser satisfactoria
C) Congelar el modelo de análisis y hacer una copia para continuar con el diseño	Se tienen dos modelos pero no van al mismo ritmo
D) Mantener los dos modelos separados	Se tienen dos modelos al mismo ritmo, pero hay una sobrecatga de mantenimiento

Modelo de diseño

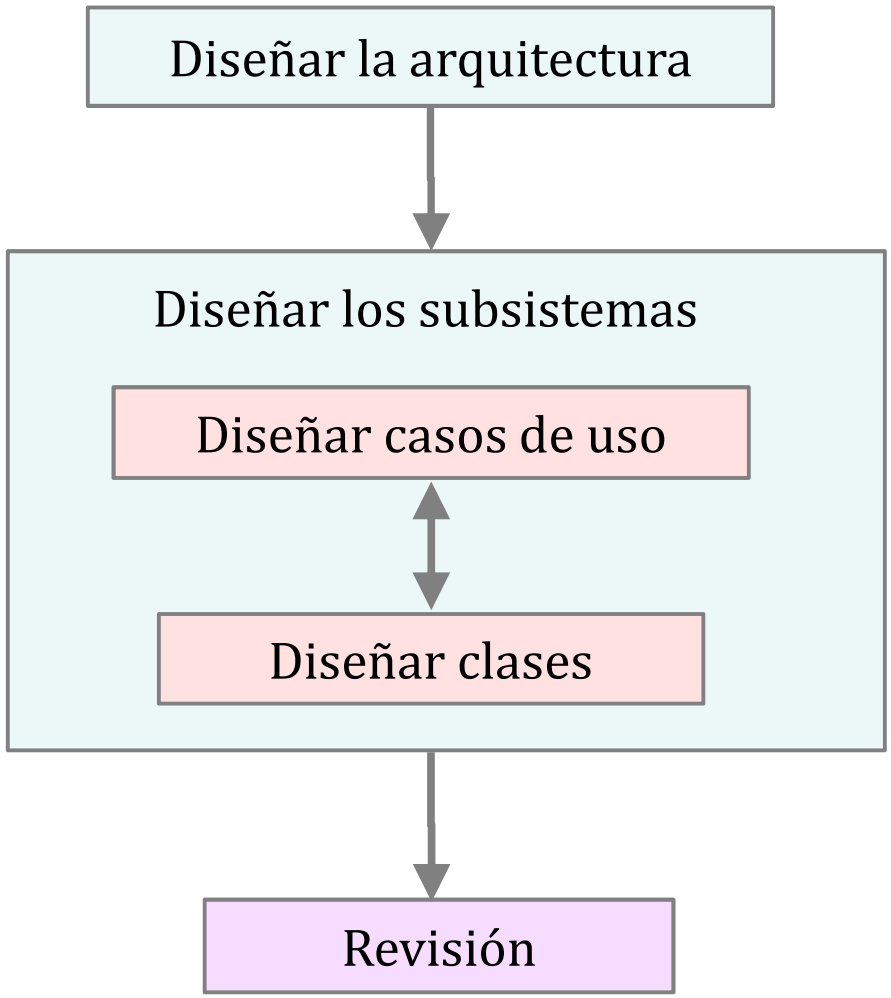
Correspondencia entre los componentes del modelo de análisis y del diseño



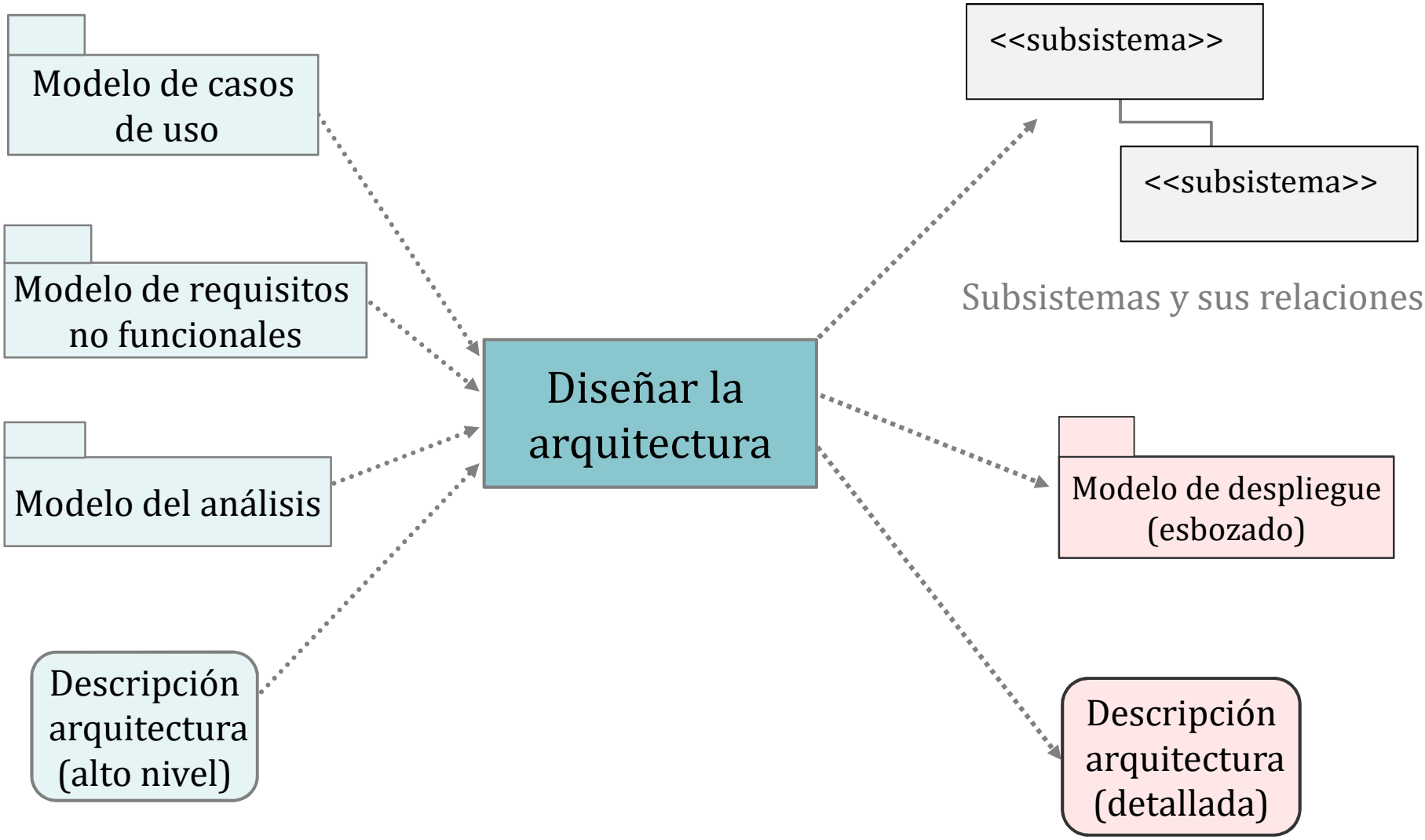
Modelo de análisis

Modelo de diseño

Tareas del diseño

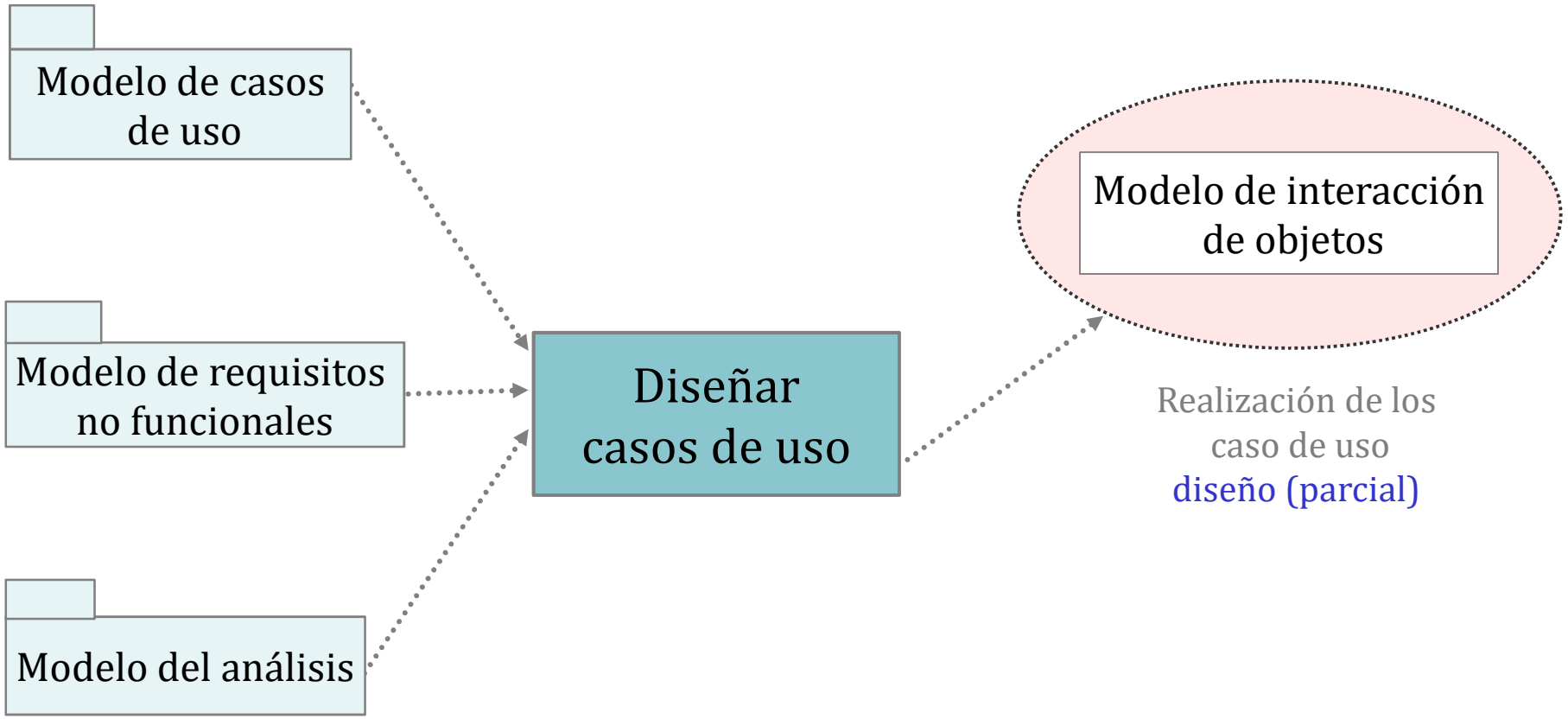


Tareas del diseño



Tareas del diseño

Para cada subsistema



Tareas del diseño