Introducción a la ingeniería de software – Clase 4

En la clase anterior ...

Ingeniería de requerimientos
Etapas.
Características de la especificación.
Herramientas de modelado.

Temas de hoy

Diseño
Definiciones.
Conceptos de diseño.

Diseño arquitectónico
Estilos. Patrones.

Buenas prácticas
Conceptos.

1

Diseño

Definiciones. Características.

Caso: diseño de una casa familiar

Varias propuestas

- Moderna con dormitorios con el espacio justo, pero amplios espacios para juegos y para “estar”
- De campo, con habitaciones amplias y galerías, orientada al exterior
- Con habitaciones para cada integrante de la familia, todas espaciosas con lugar para juegos y para estudio

Caso: diseño de una casa familiar

- Elegida la propuesta se elaboran diseños detallados
  - Para los propietarios, para que visualicen cada espacio, los tamaños, los espacios de guardado, las aberturas, ...
  - Para los obreros y técnicos: información de distribución de cañerías, cableado, soporte de las estructuras, tipos de materiales, ...

Diseño de software

- El diseño de un sistema también puede incluir
  - Diseño conceptual: concentrado en las funciones
  - Diseño técnico: describiendo la forma que tomará el sistema

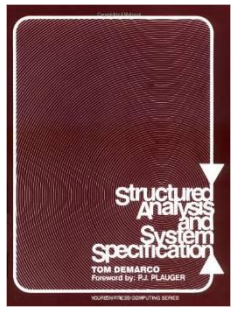
- Si se documentan por separado, se debe procurar su consistencia

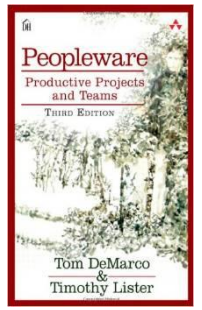
Diseñar un sistema es determinar un conjunto de componentes y de interfaces entre componentes que satisfagan un conjunto específico de requerimientos.

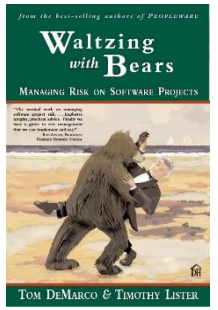
DeMarco (1982)

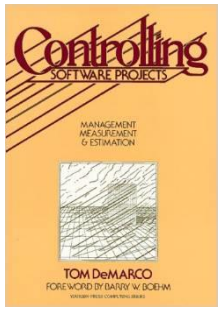
Tom DeMarco

- (1940)
- Análisis y Diseño Estructurados
- Administración de Proyectos de Software









Descomposición

- Toda descomposición separa el diseño en partes llamadas **módulos** o **componentes**.

- Un sistema es **modular** cuando una de las actividades la realiza un único componente y que además tiene bien definidas todas sus entradas y salidas.

Descomposición

- Descomposición modular: descripciones de alto nivel de las funciones de cada componente.

- Descomposición orientada por datos: basada en las estructuras externas de los datos.

- Descomposición orientada por eventos: basado en los eventos a manejar en el sistema. Cómo los eventos cambian el estado del sistema.

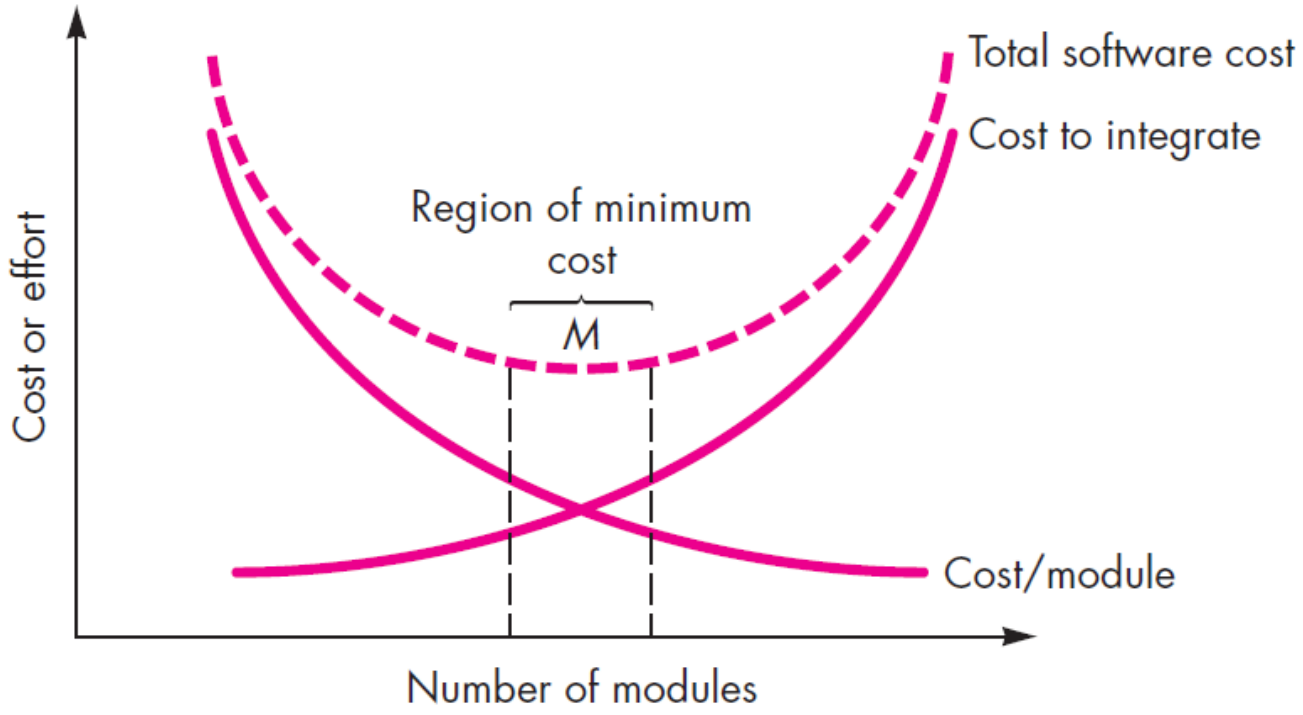
Descomposición

- Diseño “de afuera hacia adentro”: enfoque de *caja negra*. Basado en las entradas del usuario.

- Diseño orientado a objetos: identificar clases de objetos y sus interrelaciones. A nivel alto: descripción de objetos. A niveles bajos: atributos de los objetos, acciones.

Conceptos de diseño

- Modularidad
  - Componentes independientes pero integrados
  - ¿Conviene tener pocos o muchos módulos?



Conceptos de diseño

Abstracción
Abstracciones de procedimiento y de datos

Arquitectura
Estructura general y formas en las que la estructura da integridad conceptual a un sistema.

Patrones
Describe una estructura de diseño que resuelve un problema particular de diseño

División de problemas
Tiene efecto directo en el diseño y en los otros aspectos mencionados

Conceptos de diseño

Ocultamiento de la información
La información de cada módulo debe ser inaccesible a los que no necesiten de ella.

Independencia funcional
Cada módulo debe resolver un conjunto específico de requerimientos.

Refinamiento
Proceso evolutivo de elaboración.

2 Diseño arquitectónico

Niveles de diseño

Diseño de arquitectura

Asocia capacidades del sistema con componentes que las implementarán. Módulos y sus interconexiones.

Diseño de código

Algoritmos y estructuras de datos más primitivas del lenguaje de programación.

Diseño ejecutable

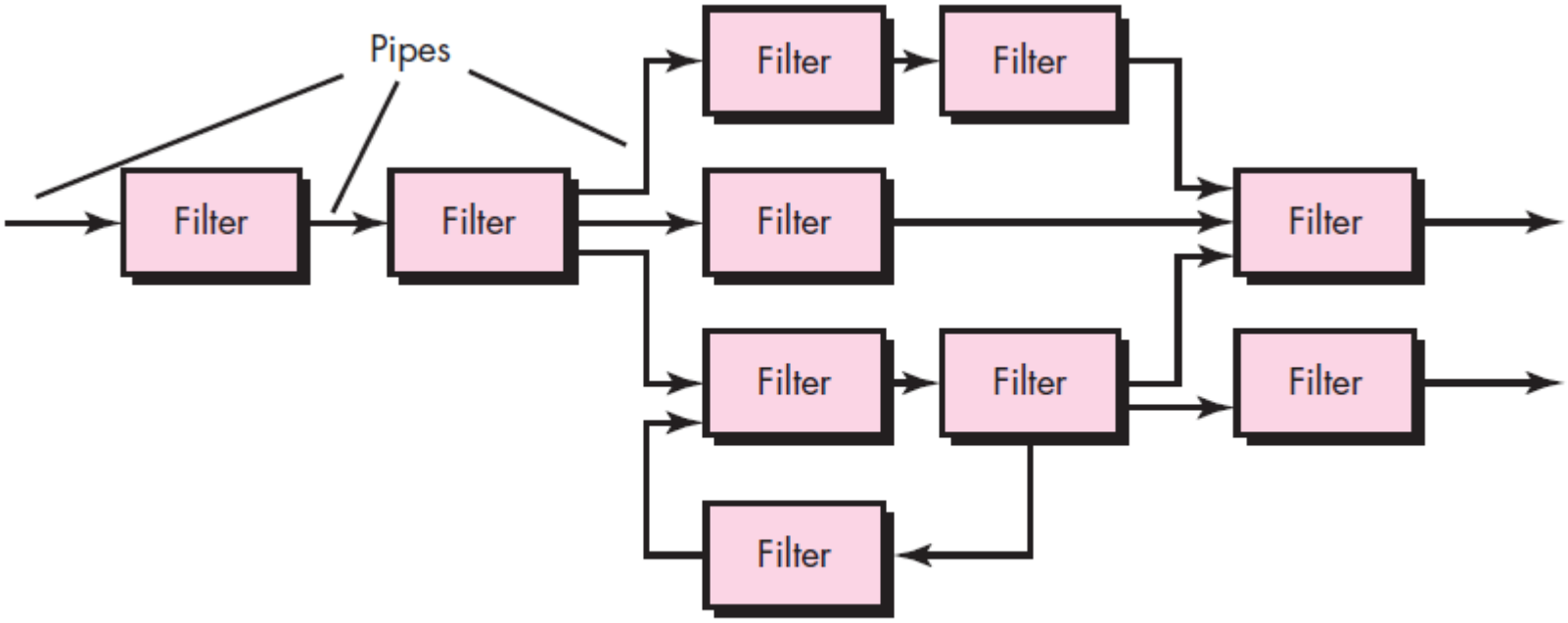
Nivel de detalle aún más bajo

Estilos de arquitectura

- Tuberías y filtros
- Diseño orientado a objetos
- Invocación implícita
- Capas
- Repositorios
- Intérpretes
- Control de procesos

Tuberías y filtros

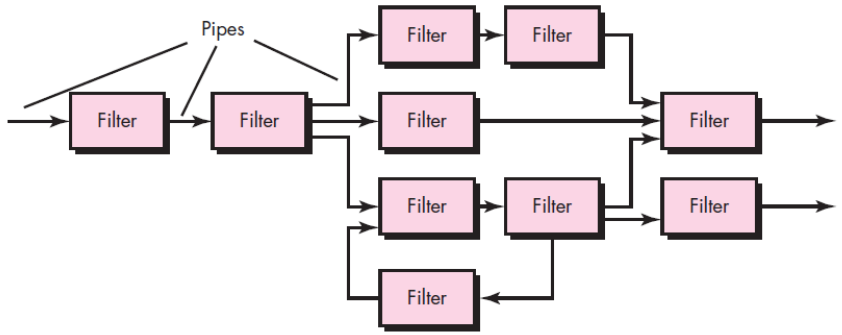
- Tuberías: corrientes de datos para entrada y salida
- Filtros: realizan las transformaciones de los datos



Pipes and filters

Tuberías y filtros

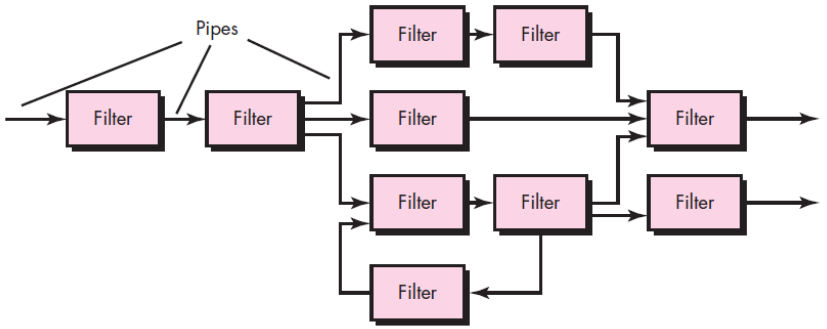
- Los filtros son independientes entre si
- Propiedades útiles
  - Comprender los efectos del sistema sobre la entrada y la salida y la composición de las transformaciones
  - Se reutilizan facilmente los filtros en otros sistemas
  - La evolución es más simple: incorporar o quitar filtros
  - Facilita la simulación del sistema para analizar propiedades
  - Ejecución concurrente



Pipes and filters

Tuberías y filtros

- Desventajas
  - Alienta el procesamiento por lotes, no es apto para aplicaciones interactivas
  - Mantener la correspondencia entre dos corrientes de datos relacionadas
  - Al ser independientes los filtros pueden duplicar funcionalidad que efectúan otros



Pipes and filters

Diseño orientado a objetos

- Es un enfoque de desarrollo de software que organiza tanto **el problema como su solución** como una colección de objetos.
- En la representación incluye tanto los datos (y su estructura) como el comportamiento.
- Esto hace que en muchos casos la descripción del problema, y el diseño de la solución sea muy similar o casi idéntica.

Diseño orientado a objetos

- En estos casos la especificación de requerimientos puede convertirse en los primeros pasos del diseño de la solución.
- Características
  - El objeto preserva la integridad de la representación de los datos
  - La representación puede ocultarse a los restantes objetos del sistema (encapsulación)

- Para representar el diseño de un sistema:
  - Identificar clases y objetos
  - Reconocer detalles de cada objeto, atributos y comportamiento
  - Identificar interacciones y relaciones entre objetos: asociaciones, composiciones, agregaciones y relaciones de herencia.

Invocación implícita

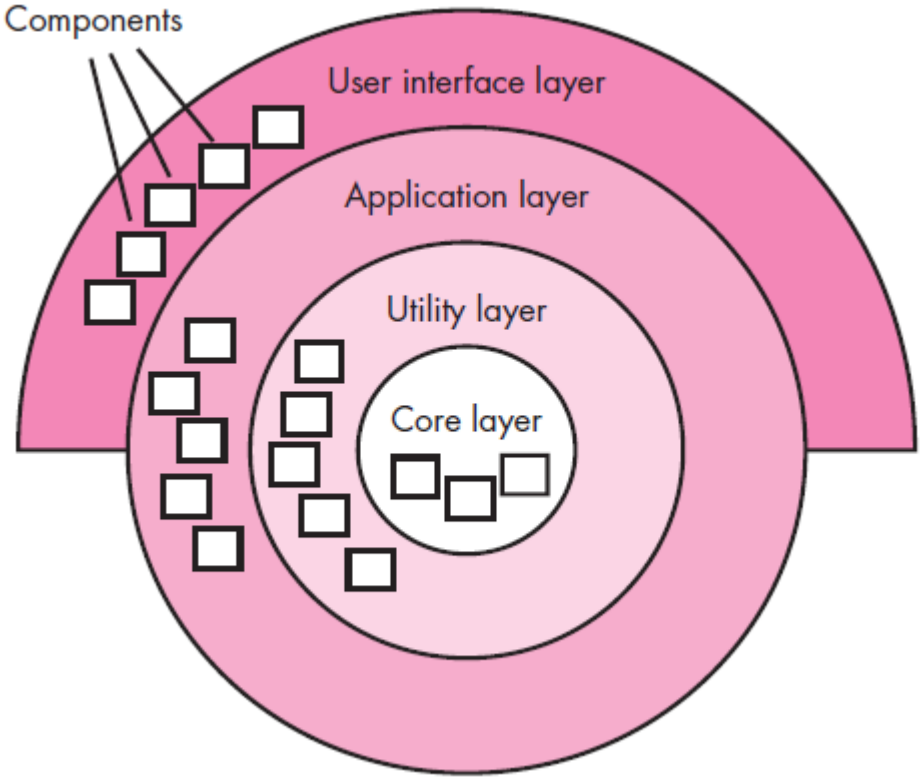
- Determinado por eventos

1. Un componente anuncia que se ha producido uno (o más) eventos
2. El sistema invoca a los procedimientos registrados

- El intercambio de datos se hace por medio de datos compartidos en un repositorio

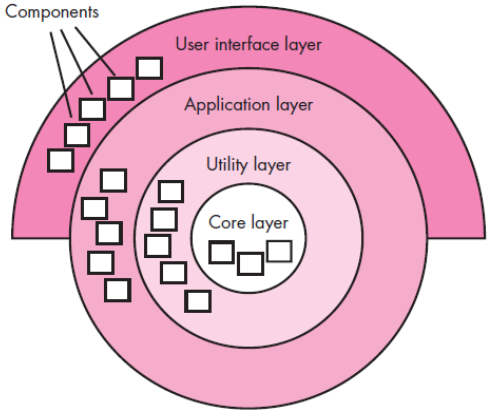
Capas

- Los modelos estratificados tienen organización jerárquica
- Cada capa:
  - Presta servicios a la inmediata superior
  - Es cliente de la inferior
- El diseño incluye los protocolos de interacción



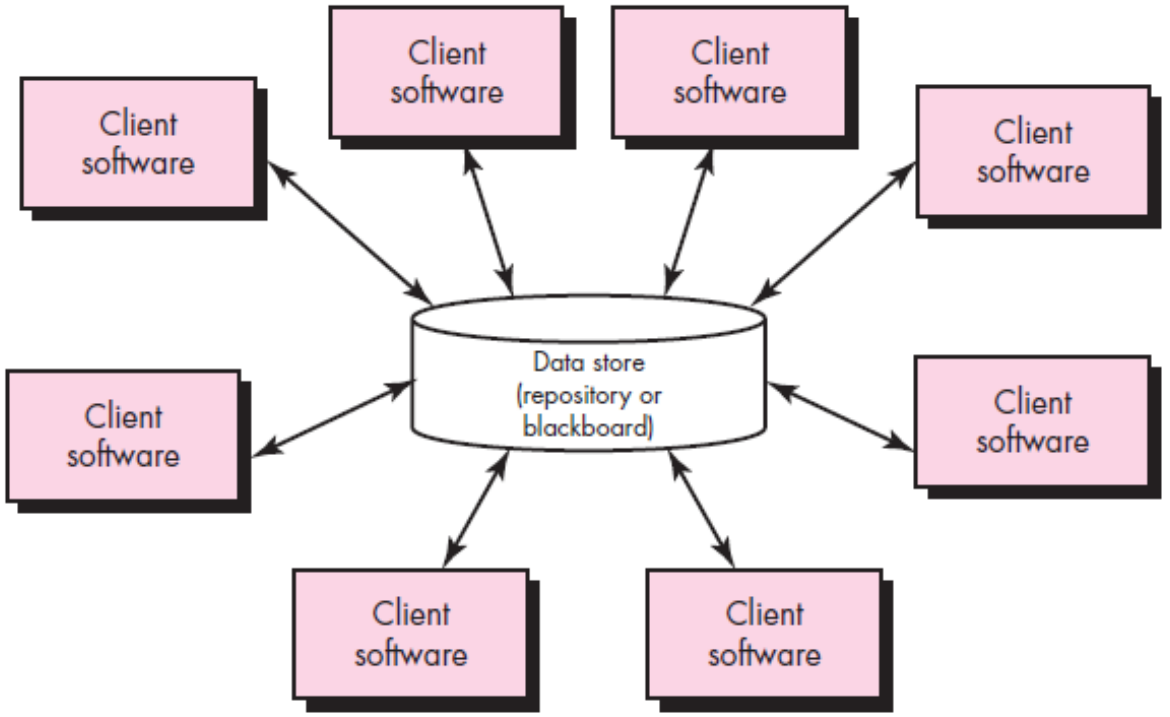
Capas

- Niveles superiores tienen mayor abstracción
- Ventaja: agregar o modificar capas es sencillo ya que el cambio solo afecta a las capas adyacentes
- Desventajas
  - No siempre es fácil estructurar sistemas de esta manera
  - Se agrega el costo de la coordinación de las capas



Repositorios

- Almacenamiento central de datos
- Conjunto de componentes que operan sobre el almacenamiento: almacenar, recuperar, actualizar
  - Ventaja: disponibilidad de los datos
  - Desventaja: el formato debe ser acceptable para todos los clientes que lo utilizan



Intérpretes

- Toma una cadena de caracteres (pseudocódigo) y la convierte en código que se ejecuta

- Permiten transformar cualquier clase de codificación en una forma más explícita y usable

Control de procesos

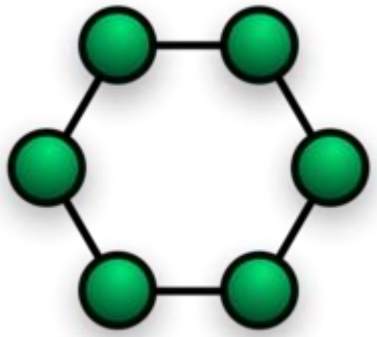
- Sistemas de propósito específico: mantener propiedades específicas de las salidas del proceso. Mantenerlas dentro o cerca de valores de referencia llamados puntos fijos o valores de calibración.
  - Por ej: monitorear proceso de fisión en una planta nuclear, la temperatura y el flujo de agua.
- Se caracterizan por las relaciones entre los componentes además de sus tipos.

Control de procesos

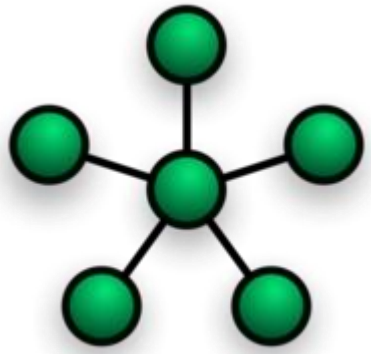
- Dos tipos de lazos de control: retroalimentación y prealimentación.
  - Retroalimentación (feedback): mide la variable y controla
  - Prealimentación (feedforward): intenta anticipar los futuros efectos sobre la variable controlada

Otros estilos: distribución

- Sistemas distribuidos
- Usualmente se describen en términos de la topología de configuración
- Anillo



- Estrella



Modularidad y abstracción

- En una descomposición modular jerárquica, los componentes de un nivel refinan los conceptos del nivel superior.
- Los niveles más “bajos” tienen mayor detalle y los niveles más “altos” cuentan con mayor abstracción.
- La abstracción permite comprender el problema planteado y la solución propuesta por el diseño.
- La modularidad permite ocultar detalles

Modularidad y abstracción

- Cada componente que oculta información, oculta decisiones de diseño.
- Los cambios en las decisiones de diseño tendrán menor impacto si hay ocultamiento de información.
- La abstracción y la descomposición modular permiten obtener diferentes vistas del sistema según el nivel en el que se ponga el foco

Diseño de interfaz

- Interfaz de usuario
  - Metáforas: términos, imágenes, conceptos a ser reconocidos y aprendidos
  - Modelo mental: organización y representación de datos, funciones, tareas y roles
  - Reglas de navegación para el modelo
  - Aspecto: características de presentación. También transmiten información
  - Sensación: técnicas de interacción para la experiencia atractiva del usuario

3 Buenas prácticas

Características de un buen diseño

- Se intenta que cada componente del diseño sea lo más independiente del resto posible
  - Mejora la comprensión de cada componente individual
  - Es más sencillo de modificar una característica de un componente si no afecta a los demás
  - Es más fácil identificar fallas
- Para medir la independencia de los componentes usamos dos conceptos
  - Acoplamiento
  - Cohesión

Acoplamiento

- Dos componentes están altamente acoplados cuando existe mucha dependencia entre ellos.
- Los componentes poco acoplados tienen pocas dependencias y/o las interconexiones son débiles.

Acoplamiento

- El acoplamiento depende de
  - Las referencias hechas de un componente a otro
  - La cantidad de datos pasados entre componentes
  - El grado de control de un componente sobre otro
  - La complejidad de la interfaz entre los componentes

Acoplamiento

- Hay distintos tipos de acoplamientos, unos menos convenientes que otros.

- Cuando un componente A modifica a un componente B, este es completamente dependiente del que lo modifica. Se denomina **acomplamiento de contenido**.
  - A modifica el valor de un item de B
  - A ramifica su ejecución en B

Acoplamiento

- Cuando un almacenamiento de datos es común a dos o más módulos se tiene **acoplamiento común**.
- Cuando un componente pasa parámetros de control a otro se dice que hay **acoplamiento de control**.
- Cuando se utiliza una estructura de datos compleja para pasar información de un módulo a otro se dice que hay **acoplamiento por estampado** (o de molde).
- Si solo comparten datos es **acoplamiento de datos**.

Acoplamiento

- de contenido
- común
- de control
- por estampado / molde
- por datos
- no acoplado

Cohesión

- La cohesión se refiere al grado de adhesión (unión) interna que tiene el componente.
- A mayor grado de cohesión, más relacionadas entre si están sus partes internas
- Un componente es cohesivo si todos sus elementos están orientados a realizar una única tarea y son esenciales para llevarla a cabo

Cohesión

- La **cohesión coincidental** ocurre cuando las partes no tienen relación alguna entre si.
- En la **cohesión lógica** los elementos están relacionados lógicamente. Ej: todas las funciones sirven para la entrada de datos, sin importar el origen.
- Cuando las funciones de un componente están relacionadas por el momento en el que ocurren o son invocadas, se trata de **cohesión temporal**.

Cohesión

- En ocasiones las funciones se agrupan únicamente porque están relacionadas por un procedimiento, hay un orden de ejecución. En este caso se dice que el componente tiene **cohesión de procedimiento**.
- Cuando un componente asocia funciones que trabajan (operan, producen) en el mismo conjunto de datos, tiene **cohesión comunicativa**.

Cohesión

- Si la relación entre las funciones es que una produce la salida que sirve de entrada a la siguiente, la **cohesión** es **secuencial**.
- La cohesión ideal es la **cohesión funcional**. En estos componentes las partes están relacionadas por la función que realizan: es una única función y todas las partes son esenciales para realizarla.

Cohesión

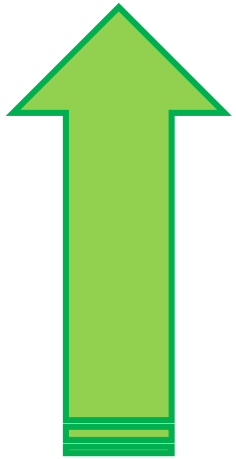
- Funcional
- Secuencial
- Comunicacional
- Procedimental
- Temporal
- Lógica
- Coincidental

Buenos diseños

- Un buen diseño modular minimiza el acoplamiento y maximiza la cohesión

Acoplamiento

Cohesión



Diseño de situaciones no deseadas

- La especificación de requerimientos dice lo que debe hacer el sistema. No explicita lo que se supone que no va a ocurrir.
- ¿Qué pasa si ocurre algo no planificado?
- Identificar lo que conocemos como **excepciones**: situaciones que se sabe que no son las esperadas.
- El diseño puede incluir **manejo de excepciones** de modo que el sistema pueda reaccionar de manera satisfactoria ante un evento de este tipo.

Excepciones

- Algunas comunes:
  - Fracasar al proporcionar un servicio
  - Proporcionar un servicio o datos erroneos
  - Corrupción de datos
- Accciones a tomar
  - Reintentar: restaurar el sistema y probar nuevamente
  - Corregir: restaurar el sistema, modificar algún aspecto y probar nuevamente
  - Informar: restaurar el sistema e informar de lo ocurrido sin proporcionar el servicio

Excepciones

- Lanzamiento Ariane 5 (501)

Excepciones

- Incidente Ariane 5 (501)
  - Menos de 40 segundos
  - Reuso de código de Ariane 4
  - Motor más rápido y potente genera números más grandes
  - Se produce “overflow” y falla de la computadora de vuelo seguido de falla de la computadora principal.
  - La falla generalizada le da demasiada potencia y se desintegra en segundos.
  - ~370 millones de USD en pérdidas

Diseño de situaciones no deseadas

- Errores humanos introducen defectos en el software.
- Si el defecto es en el diseño se puede propagar a los demás productos del proceso.
- **Falla**: desvío del sistema de su comportamiento requerido. Es percibida por el usuario.
- El **defecto** origina la falla. El defecto solo lo puede percibir el desarrollador.
- No todo defecto deriva en falla: las condiciones pueden no darse nunca.

Técnicas para mejorar el diseño

- Reducción de la complejidad
  - Reducir la complejidad desde que se detecta en los modelos.
- Diseño por contrato
  - Considera el sistema como un conjunto de componentes que se comunican.
  - Las comunicaciones están especificadas en contratos que deben definir exactamente lo que hace cada uno y qué esperan de cada uno.
  - No pueden garantizar exactitud pero dan base para pruebas y validaciones

Técnicas para mejorar el diseño

- Prototipado
  - Ofrece las ventajas que aporta en la etapa de requerimientos: mayor visibilidad, pronta disponibilidad
  - Permite explorar áreas de incertidumbre

Resumen

- Diseño. Descomposición
- Diseño arquitectónico. Patrones.
- Modularidad. Abstracción. Acoplamiento. Cohesión.
- Implementación. Codificación.
- Buenas prácticas.

Bibliografía

- *Ingeniería de software . Teoría y Práctica* - S. L. Pfleeger

  Capítulo 5 – Diseñando el sistema.

  Capítulo 6 – Considerando objetos.

- *Ingeniería del software. Un enfoque práctico* - R. Pressman

  Capítulo 8 – Conceptos de diseño.

  Capítulo 9 – Diseño de la arquitectura

  Capítulo 10 – Diseño en el nivel de componentes