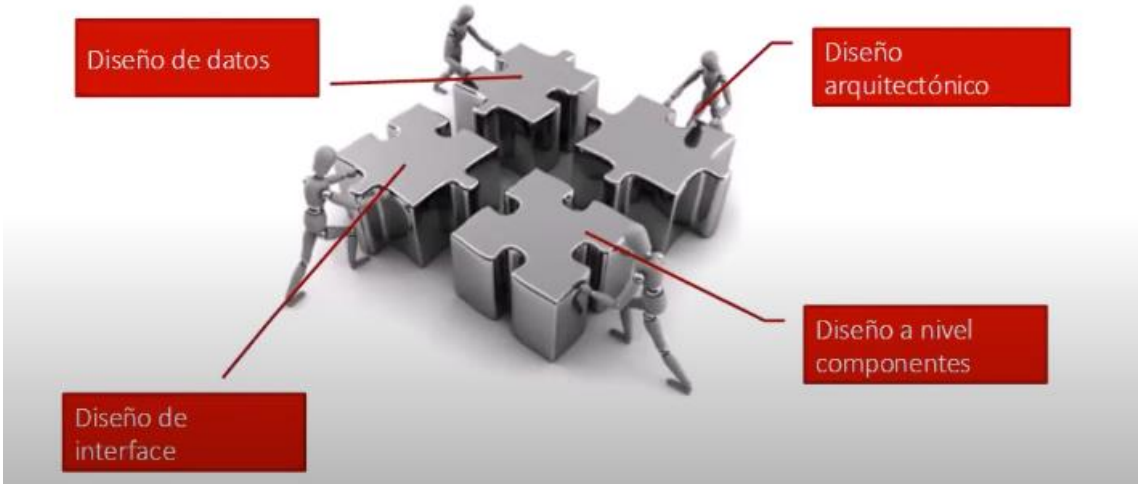
INGENIERÍA DE SOFTWARE 2 – CLASE 6: CONCEPTOS DE DISEÑO

Diseño de Software

- ¿Qué es?
  - Representación de algo que se va a construir.
  - Es donde los requisitos, necesidades y consideraciones técnicas son unidas.
  - Es la primer actividad técnica a realizar después de la etapa de análisis.
  - Es independiente del modelo de proceso utilizado.
- ¿Por qué nos importa?
  - Es el núcleo técnico con el cual de un problema llegamos a una solución mediante una transformación.

Diseño de Software – Tipos



- Diseño de datos:
  - Transforma el modelo del dominio obtenido en el análisis en estructuras de datos, objetos de datos, relaciones, etc.
- Diseño arquitectónico:
  - Define la relación entre los elementos estructurales más importantes del software, vamos a establecer estilos arquitectónicos a usar, elegimos patrones de diseño a usar, etc.
- Diseño a nivel de componentes:
  - Transforma los elementos estructurales de la arquitectura de software en una descripción procedimental de los componentes del software.
- Diseño de interface:
  - Describe la forma de comunicación dentro del mismo sistema, con otros sistemas y con las personas.

Características para evaluación del diseño

INGENIERÍA DE SOFTWARE 2 – CLASE 6: CONCEPTOS DE DISEÑO

- Deberá implementar todos los requisitos explícitos del modelo de análisis y ajustarse a todos los requisitos implícitos que desea el cliente.
- Deberá ser una guía legible y comprensible para los programadores y para aquellos que dan soporte al software.
- Deberá proporcionar una imagen completa del software.

Criterios técnicos para un buen diseño

1. deberá presentar una estructura arquitectónica que:
   Se haya creado mediante patrones de diseño reconocibles, que esté formado por componentes con buen diseño y se implemente en forma evolutiva.
2. deberá ser modular.
3. deberá contener distintas representaciones.
4. deberá conducir a estructuras de datos adecuadas y que procedan de patrones de datos reconocibles.
5. deberá conducir a componentes que presenten características funcionales independientes.
6. deberá conducir a interfaces que reduzcan la complejidad de las conexiones entre los módulos y con el entorno externo
7. deberá derivarse mediante un método repetitivo y controlado por la información obtenida durante el análisis de los requisitos del software.
8. deberá representarse por medio de una notación que comunique de manera eficaz su significado.

Importancia de los conceptos de diseño

- Los conceptos de diseño nos ayudan a:
  - Elegir criterios usados para dividir el software en componentes individuales.
  - Como extraer detalles de una función o la estructura de datos de la representación conceptual del software.
  - Identificar cuales son los criterios aplicables a todo software que definen la calidad técnica del diseño.

Conceptos de diseño

- Abstracción.
- Arquitectura.
- Patrones.
- Modularidad.
- Ocultamiento de información.
- Independencia funcional.
- Refinamiento.
- Refabricación.

INGENIERÍA DE SOFTWARE 2 – CLASE 6: CONCEPTOS DE DISEÑO

Abstracción

- Permite concentrarse en un problema a un nivel de generalización sin tener en cuenta detalles de bajo nivel.
- Tipos de abstracción:
  - Procedimental: secuencia "nombrada" de instrucciones que tienen una funcionalidad específica.
  - Datos: colección "nombrada" de datos que definen un objeto real.

Arquitectura del software

- Estructura organizacional de los componentes de un programa y ver como interactúan dichos componentes e inclusive como interactúan con las estructuras de datos que utilizan.
  - "Estructura general del software y las formas en que la estructura proporciona una integridad conceptual para un sistema.

Patrones

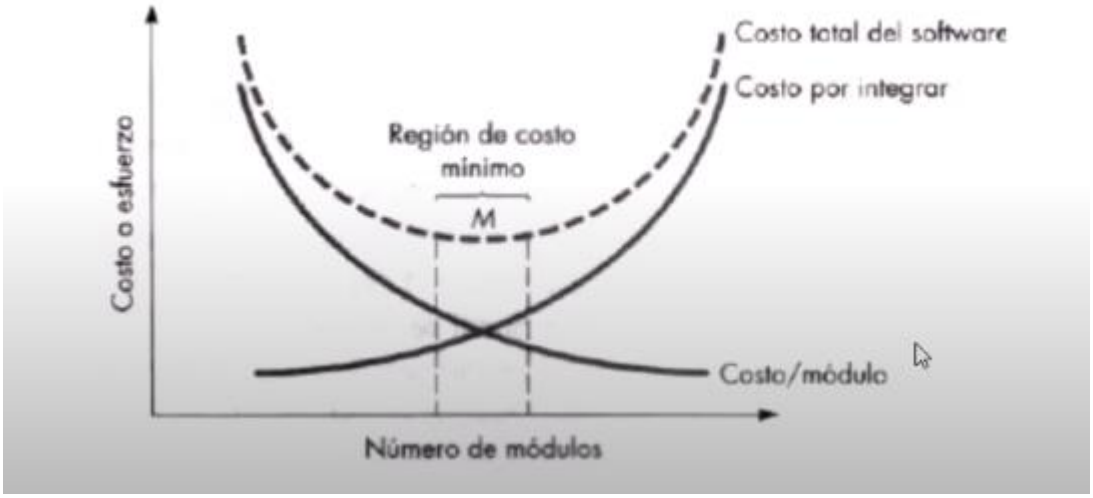
- Son una solución para un problema recurrente dentro de un contexto dado. Es una solución que ha sido demostrada y validada.
- Dicha solución describe una estructura de diseño que resuelve el problema.
- Los patrones deben proporcionar una descripción que nos permita determinar:
  - Si es aplicable al trabajo.
  - Si se puede reutilizar.
  - Si puede servir como guía para desarrollar un patrón similar pero diferente en cuanto a la funcionalidad o estructura.

Modularidad

- Es cuando el software se divide en componentes nombrados y abordados por separado, llamados frecuentemente módulos, dichos módulos se integran para satisfacer los requisitos del problema.
  - Código monolítico: una sola función que hace todo.
  - Modularización excesiva: tenemos un módulo por cada instrucción, ya es un extremo.

INGENIERÍA DE SOFTWARE 2 – CLASE 6: CONCEPTOS DE DISEÑO

- Cuantos módulos debe tener un programa?



Ocultamiento de información

- La información dentro de un modulo es inaccesible a otros que no la necesiten.
  - Un buen módulo cumple esto.

Independencia funcional

- El concepto de independencia funcional está dado por la unión de tres conceptos:
  - Modularidad + Abstracción + Ocultamiento de información.
- 

  - Cohesión: buscamos una alta cohesión.
  - Acoplamiento: buscamos un bajo acoplamiento.

Cohesión (Coherente):

- Un módulo tiene una alta cohesión cuando lleva a cabo una sola tarea y requiere poca interacción con otros módulos.
- Un módulo tiene una baja cohesión cuando lleva a cabo varias tareas diferentes sin relación entre sí.
- Hay diferentes tipos de cohesión:

INGENIERÍA DE SOFTWARE 2 – CLASE 6: CONCEPTOS DE DISEÑO

- Coincidental: cuando las sentencias dentro de un módulo llevan a cabo tareas que no tienen relación entre sí.
- Lógica: cuando las sentencias se relacionan lógicamente.
- Temporal: cuando las sentencias dentro de un módulo se deben ejecutar en un cierto intervalo de tiempo.
- Procedimental: cuando las sentencias se deben ejecutar en un orden específico.
- Comunicacional: cuando los elementos dentro de ese módulo tienen una relación con datos de entrada y salida.
- Funcional: cuando las sentencias dentro de un módulo realizan una única función. (la mejor)

Acoplamiento

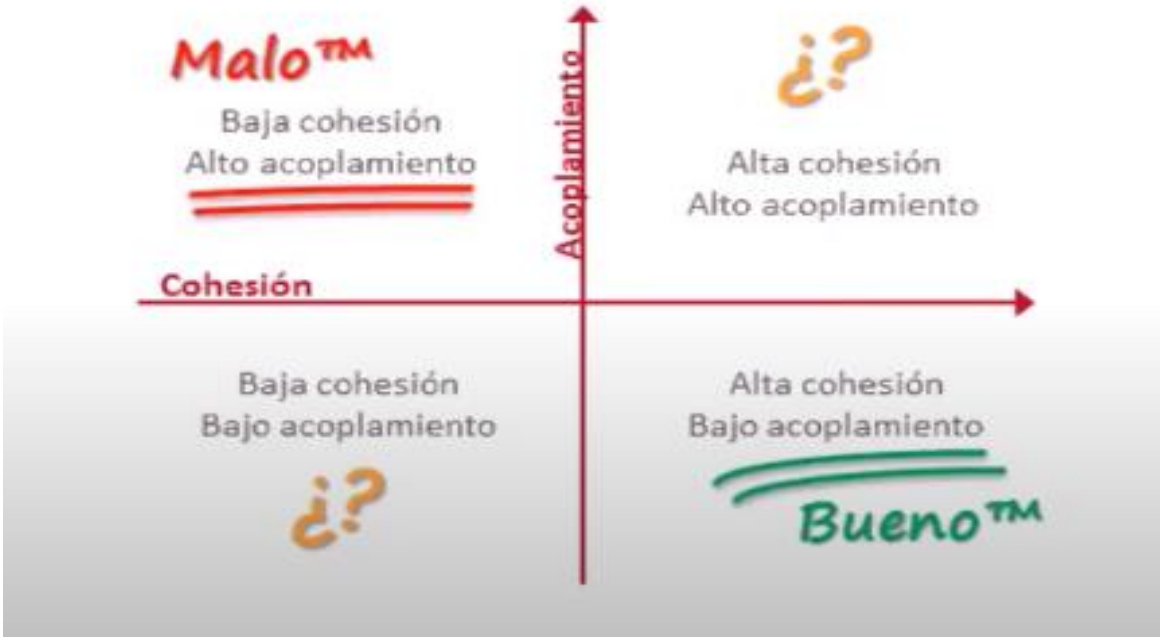
- Es la medida de interconexión entre módulos.
- Que datos interactúan, que datos van de un módulo a otro, comunicación.
    - Punto donde se realiza la entrada y los datos que pasan a través de la interfaz.
- Vamos a tratar que nuestro acoplamiento sea BAJO.

INGENIERÍA DE SOFTWARE 2 – CLASE 6: CONCEPTOS DE DISEÑO

- Acoplamiento de datos: los módulos interactúan mediante datos.
- Acoplamiento de marca: cuando los módulos interactúan mediante una estructura.
- Acoplamiento de control: cuando los módulos interactúan mediante un indicador de control.
- Acoplamiento común: una variable global usada por varios módulos.
- Acoplamiento externo: los módulos interactúan mediante protocolos de comunicación.
- Acoplamiento de contenido: un módulo utiliza un dato que está dentro de otro módulo. (el peor)

Para obtener una buena independencia funcional

INGENIERÍA DE SOFTWARE 2 – CLASE 6: CONCEPTOS DE DISEÑO



Refinamiento

- Se refina de manera sucesiva.
- La abstracción y el refinamiento son conceptos complementarios.
- La abstracción permite especificar procedimientos y datos sin considerar detalles de grado menor.
- El refinamiento ayuda a revelar los detalles de grado menor mientras se realiza el diseño.

Refabricación o rediseño (Refactoring)

- Técnica de reorganización que simplifica el diseño de un componente sin cambiar su función o comportamiento.