

Principio de Diseño

Resumen obtenido de:

Roger S. Pressman, "Ingeniería de Software. Un enfoque práctico", quinta edición, 2002

Introducción al Diseño de Software

- ¿Qué es el diseño?
 - Representación ingenieril de algo que se va a hacer. El "plano" del software.
 - Áreas que lo conforman:
 - Datos, Arquitectura, Interfaces y Componentes.
-

Introducción al Diseño de Software

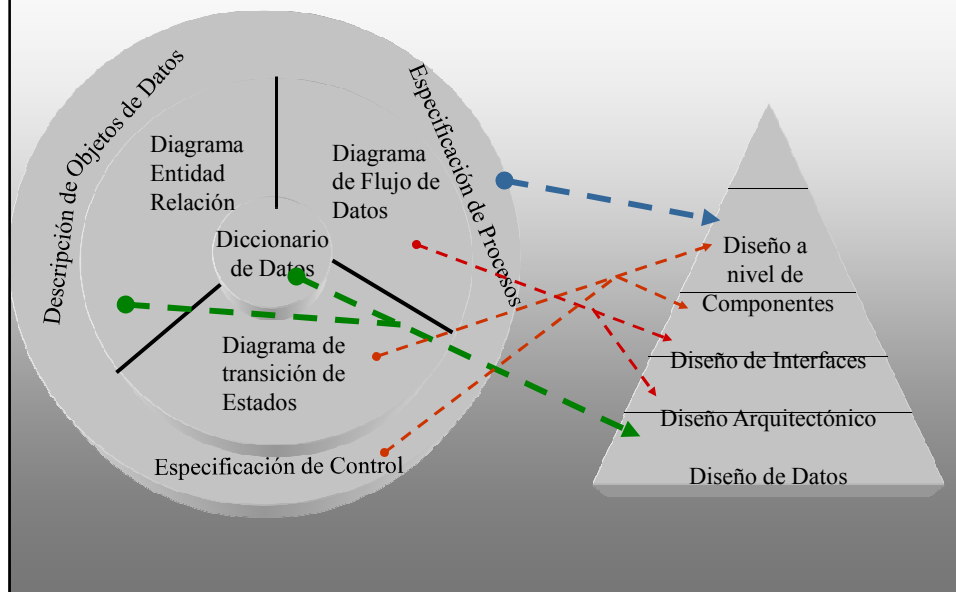
□ Resultados:

- Varios modelos para las diferentes áreas.

□ Proceso:

- Empieza con la descripción de los requerimientos y se construye, poco a poco, siguiendo los principios de diseño, hasta obtener la especificación del diseño.
-

Conversión de un Modelo de Análisis en un Diseño de Software



13.1 Diseño de Software (DS) e Ingeniería de Software

- DS se encuentra en el núcleo técnico de la IS.
 - Diseño de Datos.
 - transforma el dominio de información en estructuras de datos.
 - Diseño arquitectónico.
 - define la relación entre elementos estructurales del software.
-

13.1 Diseño de Software e Ingeniería de Software

- Diseño de Interfaz manera de comunicarse del software:
 - Dentro de sí.
 - Con otros sistemas.
 - Con las personas.
 - Diseño de Componentes.
 - Transforma elementos estructurales en descripciones procedimentales.
-

13.2 El Proceso de Diseño

- Es un proceso iterativo.
 - Inicialmente se presenta en un nivel de abstracción alto y cada vez se refina más.
 - La calidad de su evolución se evalúa mediante Revisiones Técnicas Formales (RTF).
 - Guía para la evaluación de un buen diseño:
 - Deberá implementar todos los requerimientos.
 - Ser legible y comprensible para poder codificar.
 - Proporcionar imagen completa de todos los elementos de un DS.
-

Directrices para la calidad del Diseño

- Deberá presentar una estructura arquitectónica que:
 - Creada con patrones de diseño reconocibles.
 - Formada por componentes de calidad y funcionalmente independientes.
 - Con interfaces que faciliten las conexiones.
 - Que se pueda implementar evolutivamente.
 - Ser modular.
 - Contener diferentes representaciones.
 - Derivarse del análisis de forma controlada.
-

13.3 Principio de Diseño de Software (DS)

- El DS es una secuencia de pasos, no es una receta pues intervienen:
 - La creatividad, experiencia y un compromiso con la calidad.
- Existen factores de calidad internos y externos.
 - Externos. Propiedades que pueden ser observadas por los usuarios.
 - Internas. Son buscados por el Ingeniero de Software.



13.3 Principio de Diseño (según Alan Davis)

- | | |
|---|---|
| 1. En el proceso deben tomarse enfoques alternativos. | 6. Deberá estructurarse para admitir cambios. |
| 2. Deberá rastrearse hasta el análisis. | 7. Debe prever la adaptación a circunstancias inusuales. |
| 3. Se debe reutilizar. | 8. No codificar. |
| 4. Tratar de imitar el dominio del problema. | 9. Evaluarse en función de calidad mientras está creciendo. |
| 5. Uniformidad e integración. | 10. Minimizar errores conceptuales. |
-


13. 4 Conceptos de Diseño

1. Abstracción.

- Existen diferentes niveles, cada paso del proceso software es un refinamiento.
 - Alto. En el lenguaje del entorno del problema.
 - Inferiores. Orientación procedimental.
 - Tipos de abstracción:
 - Procedimental. Secuencia nombrada de instrucciones que tiene una función específica y limitada.
 - De datos. Colección nombrada de datos que describe un objeto de datos.
 - De control. Explicita el mecanismo de control de programa sin especificar los datos internos.
-

13. 4 Conceptos de Diseño

(1. Abstracción)

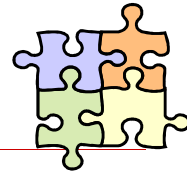
- Ejemplo de tipos de abstracción para una Modelar una puerta:
 - Procedimental. Operación: Abrir
 - Acercarse a la puerta;
 - Localizar mecanismo de apertura;
 - Accionar mecanismo de apertura;
 - Pasar por la puerta (jalar, empujar, esperar apertura).
 - De datos. La Puerta.
 - Tipo de puerta (madera, metal, etc),
 - Mecanismo de apertura, Peso, Dimensiones.
- 
-

13. 4 Conceptos de Diseño (2. Refinamiento)

- Debe hacerse paso a paso.
- El refinamiento hace que el diseñador:
 - trabaje sobre la sentencia original,
 - proporcionando cada vez más detalles
 - a medida que van teniendo lugar *sucesivamente* todos y cada uno de los refinamientos

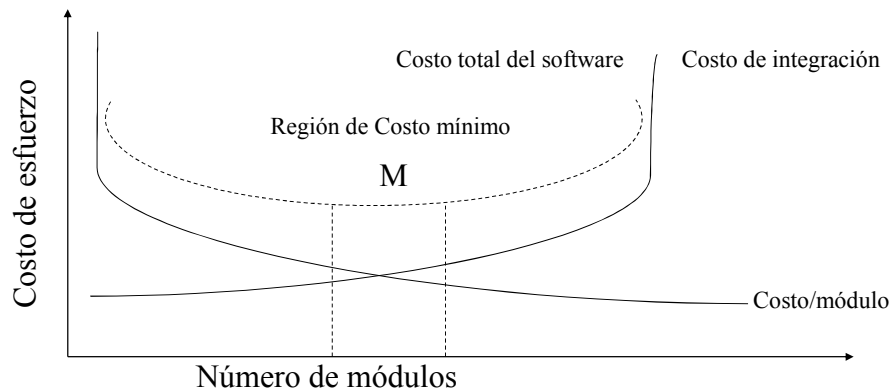
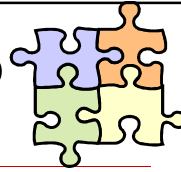


13. 4 Conceptos de Diseño (3. Modularidad)



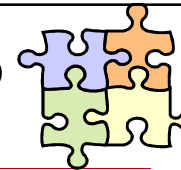
- Se trata de dividir el software en componentes nombrados y abordados por separado llamados *módulos*, que se integran para resolver los requisitos del problema.
 - Según G. Meyers, “la modularidad es el único atributo del software que permite gestionar un programa intelectualmente”
 - Un software monolítico (programa grande formado por un único módulo) no puede ser entendido fácilmente por el lector.
-

13. 4 Conceptos de Diseño (3. Modularidad)



Aunque la modularidad es buena y está basada en la premisa de “divide y vencerás”, no debe exagerarse el número de módulos

13. 4 Conceptos de Diseño (3. Modularidad)



- Criterios para evaluar una metodología de diseño respecto a su habilidad de definir un sistema modular efectivo:
 - Capacidad para descomposición modular.
 - Capacidad de empleo de componentes modulares
 - Capacidad de comprensión modular
 - Continuidad modular
 - Protección modular

13. 4 Conceptos de Diseño (4. Arquitectura de Software)

- ❑ Alude a la estructura global del software y a las formas en que ésta proporciona la integridad conceptual de un sistema.
 - ❑ Sirve como marco de trabajo desde donde se llevan a cabo actividades de diseño más detalladas.
 - ❑ Mediante patrones de arquitectura el Ingeniero de Software podrá reutilizar.
 - ❑ Existen LDA (Lenguajes de Descripción Arquitectónica) para describir la arquitectura de software.
-

13. 4 Conceptos de Diseño (4. Arquitectura de Software)

- ❑ Propiedades de un diseño arquitectónico:
 - Estructurales. Define los componentes (módulos, objetos, interfaces), la manera en que se empaquetan e interactúan.
 - Extra-funcionales. Cómo consigue los requerimientos no funcionales. Ejemplos: rendimiento, capacidad, fiabilidad, seguridad, etc.
 - Familias de sistemas relacionadas. Debe poder utilizar bloques de construcción arquitectónicos.

