

DISEÑO

- Proceso de Definir → Arquitectura
Componentes
Interfaces
Otras características del sistema/Comp.
- Y el resultado del Proceso.

DEFINICIONES -

Sistema → Una entidad lógica, que tiene un conjunto de resp. onabilidades u objetivos y consiste en HW o SW o ambos.

Sub Sistema → Un sistema que es parte de un sistema mayor
Tiene una interface bien definida.

Componente → Cualquier pieza de HW o SW que tenga definido un rol claro.

- Un comp. puede ser aislado, permitiendo que se lo reemplace con un componente diferente que tenga una funcionalidad equivalente.
- Muchos componentes están diseñados para ser reutilizables.
- Recíprocamente, otros realizan funciones con un propósito especial.

Módulo → Un componente que es definido a nivel de lenguaje de programación
→ Package / namespace.

Tareas de Aplicación 1 Clasificando la lista según los criterios de arriba.

- 1 - Sistema Operativo → Sistema
- 2 - Browser → Sistema
- 3 - Microservicio de pago → Sub sistema
- 4 - Compilador → Componente.
- 5 - Biblioteca de Criptografía → Componente.
- 6 - Controlador de dispositivo → Componente.
- 7 - Servidor web → Sub sistema
- 8 - Base de datos SQL → Componente
- 9 - Middleware → Componente
- 10 - ORM → Módulo
- 11 - IDE → Sub sistema
- 12 - Motor de Renderizado → Sub sistema
- 13 - Procesador de texto → Componente
- 14 - Gestor de notificaciones → Módulo
- 15 - Gestor de ventanas → Subsistema.

Proceso de Diseño de Software.

- **Arquitectónico** → Se define la Arq. (utiliza patrón de arg. en capas, mvc, monolítico, microservicio, SOA)
- **Especificación Abstracta** → Se especifican los subsistemas.
Cada subsistema realiza un servicio importante.
 - Contiene los objetos altamente acoplados.
 - Relativamente independiente de otros subsistemas.
 - Se descomponen en módulos y puede también en subsist. más pequeños.
- **Diseño de Interface** → Se describen las interfaces de los

Sistemas.

- Diseño de Componentes → Se descompone cada subsistema en componentes.

PRINCIPIOS DE DISEÑO

- Dividir y Conquistar → Tratar con algo grande de estructura, es normalmente más difícil que si se hace con cosas más pequeñas.
- Incrementar la cohesión → Todo lo que sea posible. Un sub-sistema o módulo tiene un grado alto de cohesión si mantiene juntas las cosas que están relacionadas y afuera las restantes.
- Reducir Acoplamiento → Todo lo que sea posible. El acoplamiento ocurre cuando existen interdependencias entre un módulo y otro.
- Nivel de Abstracción → Se debe mantener alto como sea posible. Reducir complejidad.
- Incrementar Reusabilidad → Reutilización de código y diseño cuando sea posible.
- Anticipe Obsolescencia → Planificar cambios de tecnología o el entorno, de manera que el software continúe ejecutándose.

