

# Parcial 2 - Ing. Software

¿QUÉ ES UNA ARQUITECTURA?	¿QUÉ NO ES UNA ARQUITECTURA?
Vista Estructural de alto nivel	Una normativa madura
Define estilo(s) para una solución	Diseño de Software con UML
Se concentra en requisitos no funcionales	Naturalmente vinculada a metodología

## IMPORTANCIA

- Permite entender cómo debe organizarse un sistema.
- Explica cómo tiene que diseñarse la estructura global de ese sistema.
- Es la primera etapa en el diseño de Desarrollo de Software.

## PARA QUE SIRVE

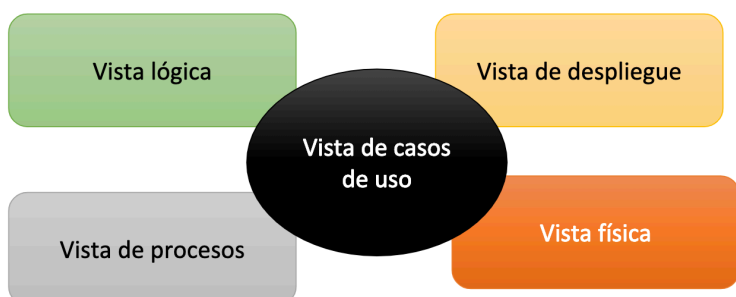
- Plan de Diseño para negociación de requisitos del sistema.
- Medio para establecer discusiones con clientes, desarrolladores, etc.

## NIVELES DE ABSTRACCIÓN DE UNA ARQUITECTURA

<i>Arquitectura en Pequeños</i>	<i>Arquitectura en Grandes</i>
Se interesa por la arquitectura de programas individuales	Se interesa por la arquitectura de programas empresariales complejos

## VENTAJA DE DOCUMENTAR UNA ARQUITECTURA

- Comunicación con los participantes.
- Análisis del sistema.
- Reutilizar a gran escala.



## VISTAS ARQUITECTÓNICAS

- Vista Lógica
- Vista de Proceso
- Vista de Desarrollo
- Vista Física
- Vista Conceptual

# Vistas Arquitectónicas

Logica

Proceso

Conceptual

Desarrollo

Física

## Vista Logica

- Soporta el análisis y las especificaciones de los requisitos funcionales.
- El sistema se descompone en un conjunto de abstracciones clave del problema.

NOTACIÓN	ESTILO
UML <i>Diagrama de Clases - Diagrama de Paquetes</i>	<i>Orientado a Objetos</i>

## Vista de Proceso

- Util para hacer juicios acerca de las características como el rendimiento y disponibilidad
- Se tratan algunos requisitos no funcionales.
- Especifica qué hilo de control ejecuta cada operación identificada en cada clase.

NOTACIÓN	ESTILO
UML <i>Diagrama de Estados, Actividades y Similares</i>	<i>Taxonomía de Garlan y Shaw</i>

## Vista Conceptual

- Corresponde con instancias que unifican todas las vistas
- Debería poder hacer una trazabilidad a todos los componentes del sistema software.

NOTACIÓN
UML <i>Diagrama de Casos de Uso</i>

## Vista de Desarrollo

- Muestra cómo se descompone el software en elementos implementados en grupo o individual.
- Se enfoca en la organización de los modelos de software en el desarrollo.
- Toma los requisitos internos relacionados con facilidad de desarrollo, gestión del software, etc.

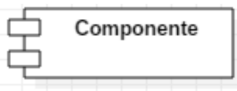
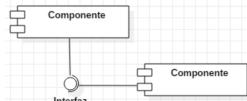
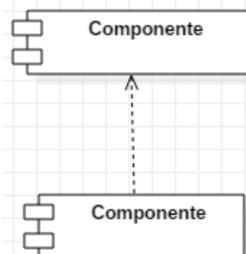
NOTACIÓN	ESTILO
UML <i>Diagrama de Componentes - Diagrama de Paquetes</i>	<i>Definir de 4 a 6 capas de subsistemas</i>

## Vista Fisica

- Expone el hardware y software del sistema y como los componentes se distribuyen
- Se centra en los requisitos no funcionales

NOTACIÓN
UML <i>Diagrama de Despliegue</i>

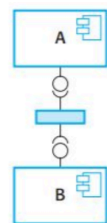
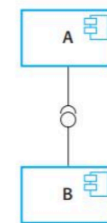
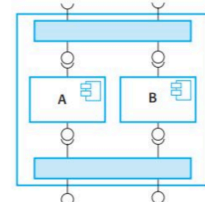
# Diagrama de Componentes

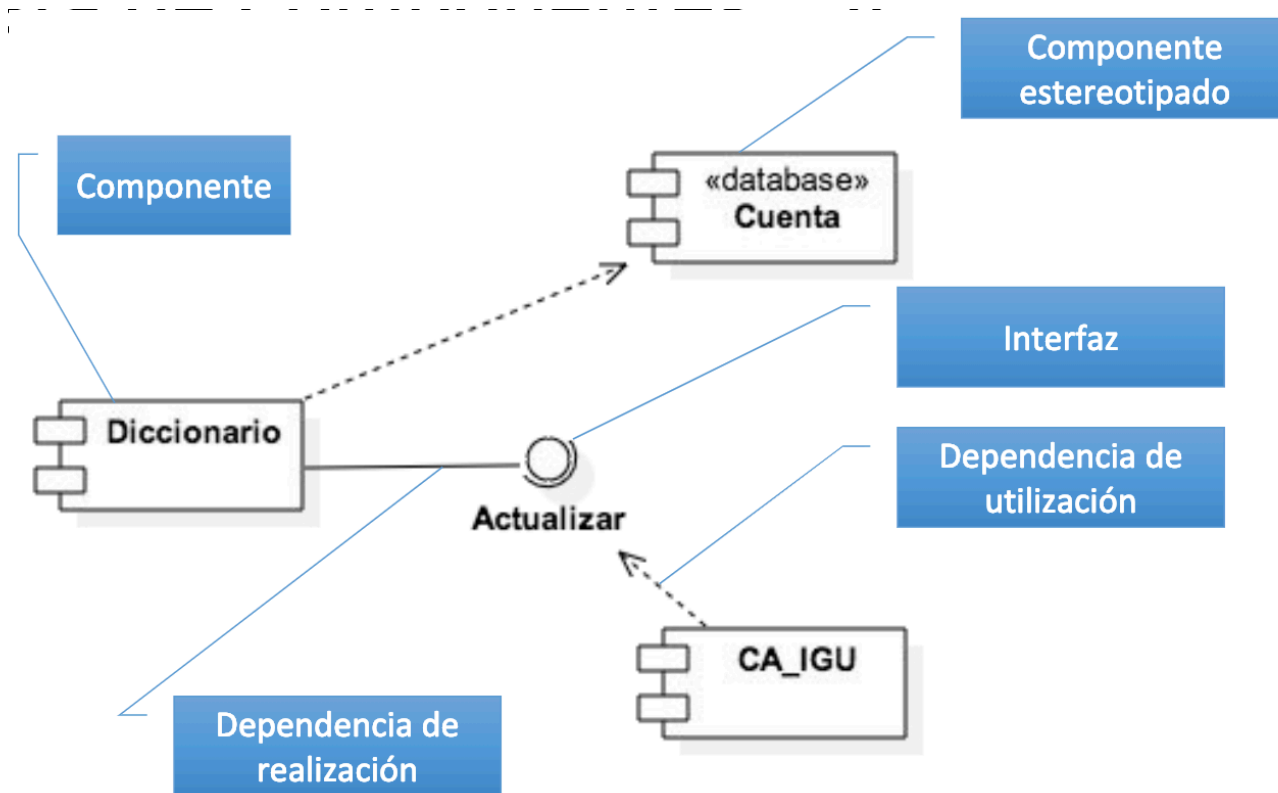
CLASE ABSTRACTA	INTERFAZ	DEPENDENCIAS
Define <i>polimorfismo</i> de objetos distintos que hacen cosas igual pero de forma diferente	Colección de Métodos abstractos y propiedades	Se muestra como una flecha punteada
	Se especifica que se debe hacer pero no su implementación	Quiere decir que un componente necesita de la otra para completar su definición
		

## COMPONENTE:

- Es una parte física y reemplazable de un sistema.
- Es la materialización de una o mas clases

## TIPOS DE COMPOSICIÓN

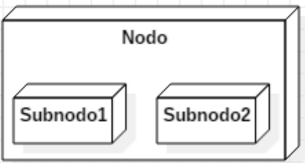


SECUENCIAL	
JERÁRQUICA	
ADITIVA	

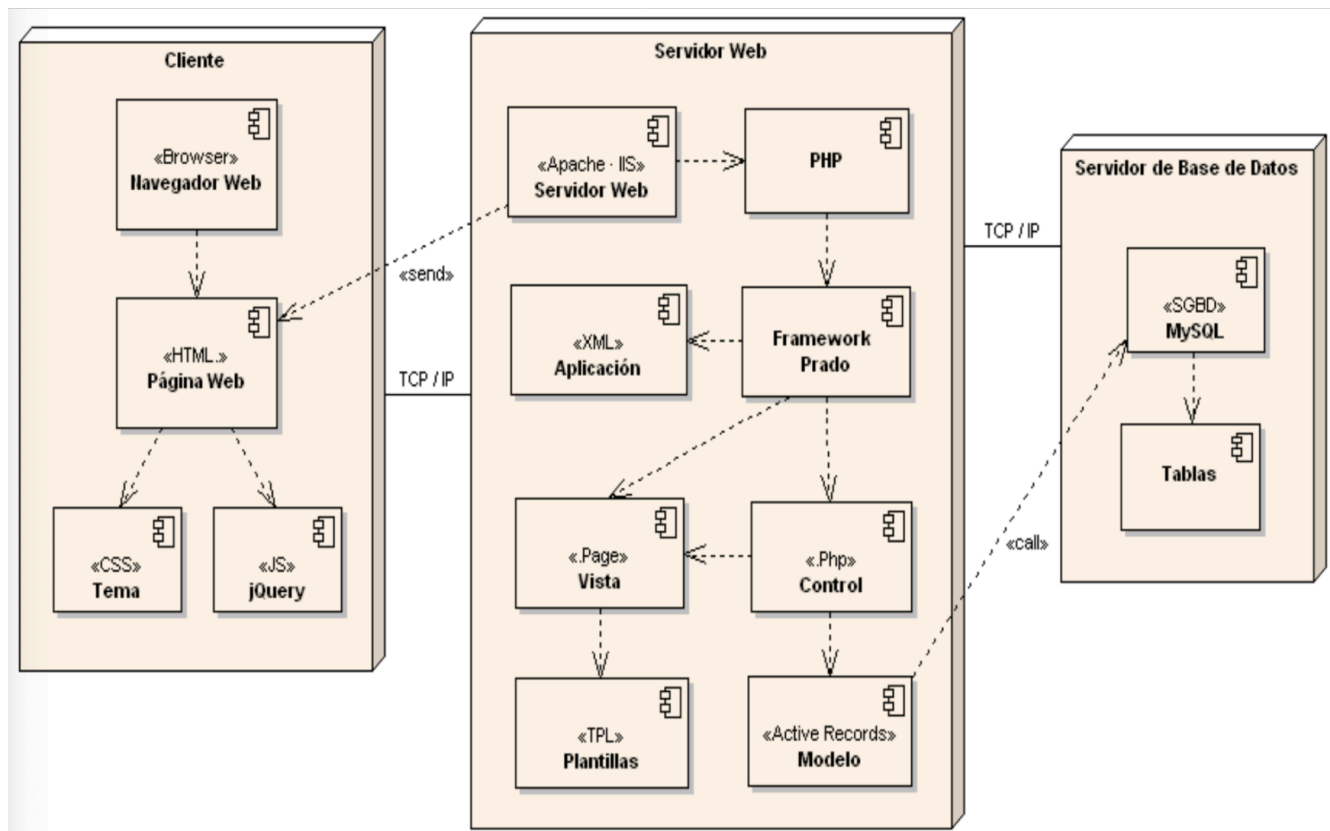


# Diagrama de Despliegue

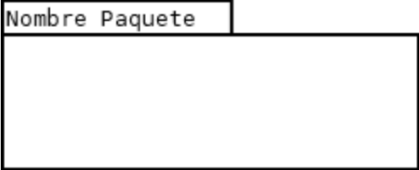
Describe la arquitectura física del sistema durante la ejecución, en términos de:

- Procesadores
- Dispositivos
- Componentes de software

NODO	DISPOSITIVOS	RELACIONES
Es un objeto físico de ejecución que representa un recurso computacional.	También se representan como nodos	Los nodos se conectan mediante asociaciones de comunicación indicando:
Pueden tener estereotipos para distinguir diferencias	Se usan estereotipos para identificar el tipo de dispositivos	<ul style="list-style-type: none"> <li>• algún tipo de ruta de comunicación entre nodos</li> <li>• Los nodos intercambian objetos o envían mensajes</li> </ul>
<ul style="list-style-type: none"> <li>• CPU</li> <li>• Memorias</li> <li>• Dispositivos</li> </ul>		
		



# Diagrama de Paquetes

PAQUETE	DEPENDENCIA	SUBSISTEMA
Refleja la arquitectura de alto nivel de un sistema	Deben verse a un alto nivel de abstracción	Es un paquete que tiene piezas separadas de especificación y realización
Es una parte de un modelo	Implica que existe un enfoque ascendente	Representa una unidad coherente del modelo, con interfaces limpias al resto del sistema
Cada parte de un modelo debe pertenecer a un paquete	Representa que un paquete necesita de los elementos de otro paquete para funcionar	
La asignación debe seguir un cierto principio racional, tal como, funcionalidad común, implementación y punto de vista.	Flecha discontinua que va desde el paquete que requiere la función hasta el paquete que ofrece esa función	Se dibujan como paquetes con las palabras clave de estereotipo
Los paquetes pueden tener otros paquetes		
 <p>Notación de un paquete</p>	