INDICE

[TEMA1: INTRODUCCIÓN A LA ARQUITECTURA DE SOFTWARE 2](#_Toc101562142)

[Arquitectura de Software 2](#_Toc101562143)

[Arquitectura de Sistemas de Software 2](#_Toc101562144)

[Ingeniero de Software 2](#_Toc101562145)

[Arquitecto de Software 2](#_Toc101562146)

[Arquitectura Vs Diseño 2](#_Toc101562147)

[Actividades Típica en el Diseño 2](#_Toc101562148)

[Arquitectura de Software y Calidad de Software 3](#_Toc101562149)

[Tipos de Requisitos 3](#_Toc101562150)

[TEMA2: PATRONES DE ARQUITECTURA DE SOFTWARE 4](#_Toc101562151)

[Estilos Arquitectónicos 4](#_Toc101562152)

[Tubos y Filtros 4](#_Toc101562153)

[ESTILO CENTRADO EN DATOS: Repositorio 5](#_Toc101562154)

[Llamada y Retorno: Arquitectura 3 capas 6](#_Toc101562155)

# TEMA1: INTRODUCCIÓN A LA ARQUITECTURA DE SOFTWARE

## Arquitectura de Software

Involucra al diseño estructural del software que forma parte de un sistema.

## **Arquitectura de Sistemas de Software**

Involucra al diseño estructural de todo el sistema, entiéndase: Software, Datos, Usuarios, el Ambiente Operacional y Procedimientos.

## Ingeniero de Software

* Se preocupa de la parte estructural, y le preocupa poco la estética.
* Sigue procedimientos bien definidos, lo suyo es más ingeniería que arte.
* Se preocupa por los costos a corto y largo plazo
* Construye en base a los requisitos funcionales.

## Arquitecto de Software

* Se preocupa de la parte estética y funcional (usabilidad).
* Interactúa mucho con el cliente.
* Construye en base a los requisitos No Funcionales’
* Utiliza ciclos cortos de trabajo
* Trabaja en grupos muy reducidos

## Arquitectura Vs Diseño

## Actividades Típica en el Diseño

* Diseño de la Base de Datos
* Diseño de la Interfaz
* Diseño de la Arquitectura de Software
* Diseño de Detalle Procedimental

## Arquitectura de Software y Calidad de Software

## Tipos de Requisitos

* **Requisitos Funcionales:** Indican ¿Qué? Deben ser alcanzados sí o sí.
* **Requisitos de Calidad:** Indican ¿Cuán bien? La calidad final dependerá del logro de estos objetivos.
* **Requisitos de Restricción:** Indican ¿Cuánto? En función de costos, tiempos, personal, etc.

# TEMA2: PATRONES DE ARQUITECTURA DE SOFTWARE

Un patrón de arquitectura de software es un esquema genérico probado para solucionar un problema particular recurrente que surge en un cierto contexto. Este esquema se especifica describiendo los componentes, con sus responsabilidades, relaciones, y las formas en que colaboran.

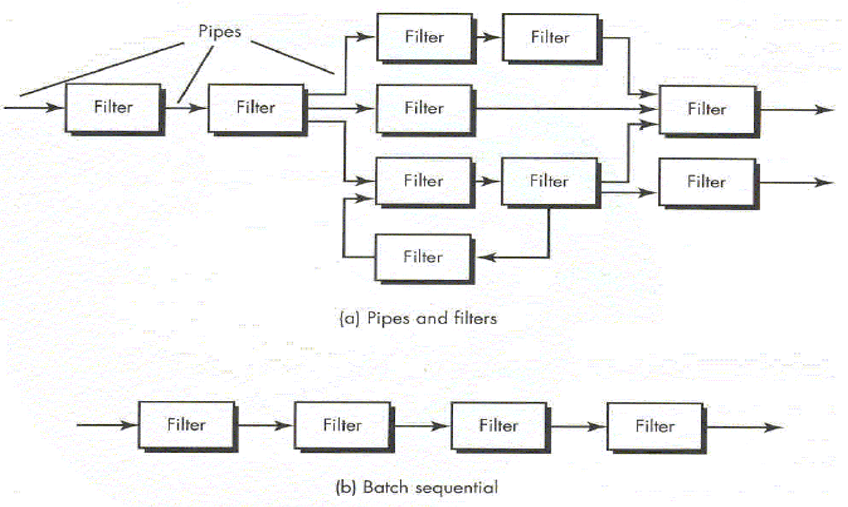
## Estilos Arquitectónicos

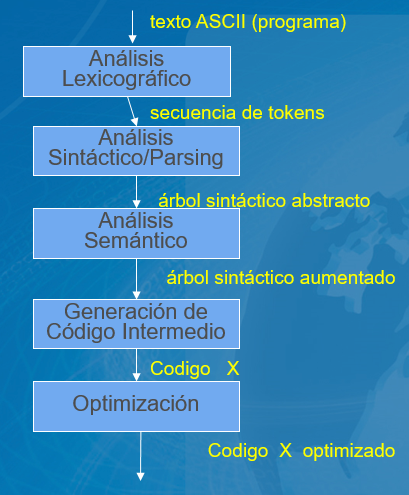
## Tubos y Filtros

El estilo de arquitectura de tubos y filtros (Pipe&Filters) provee una estructura para procesar flujos de datos.

Los Filtros (Filters) son los encargados de realizar todas las operaciones con las entradas que reciben para transformarlos en determinadas salidas

Las Tuberías (Pipes) son las encargadas de llevar el resultado que de un filtro hacia otro para que siga el proceso de transformación y se obtenga la salida final



****

**Ejemplo de Compilador**

**Ventajas**

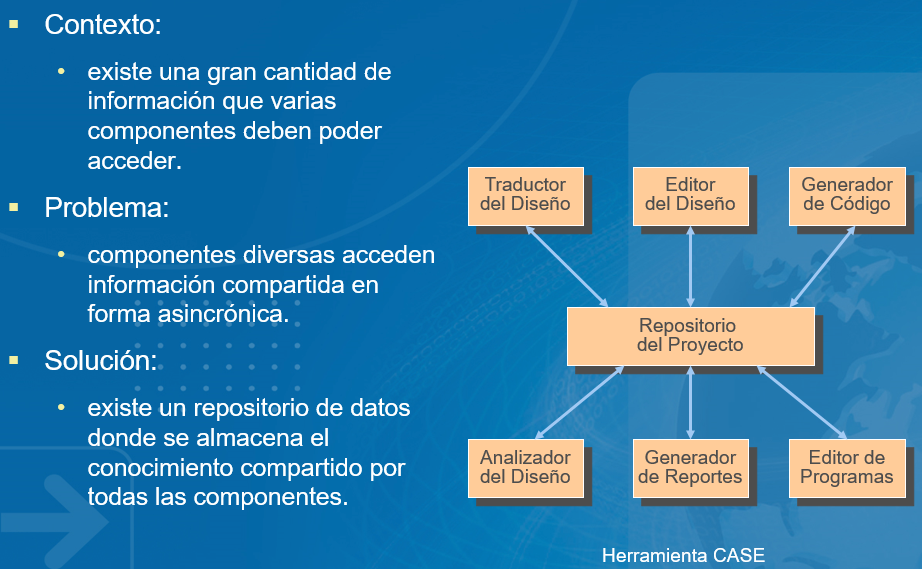
* Facilita la reutilización de transformaciones.
* Relativamente sencillo de implementar, a nivel concurrente o secuencial.
* Permite entender la E/S como composición de filtros
* Reutilización: dos filtros cualesquiera pueden unirse
* Mantenimiento: creación y reemplazo de filtros

**Desventajas**

* Requiere algún formato común para transferir los datos a través del pipeline.
* Es difícil soportar interacciones basadas en eventos.
* No muy útil para sistemas interactivos.

## ESTILO CENTRADO EN DATOS: Repositorio

En esta arquitectura hay dos componentes principales: una estructura de datos que representa el estado actual y una colección de componentes independientes que operan sobre él.



**Ventajas**

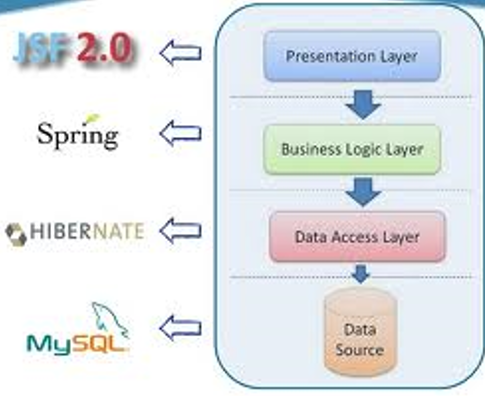
* Forma eficiente de compartir grandes cantidades de información.
* No hay necesidad de transmitir información entre las distintas componentes del sistema.
* Las actividades de respaldo, seguridad, control de acceso y recuperación de errores están centralizadas.

**Desventajas**

* Todas las componentes deben acordar un formato para los datos del repositorio.
* El rendimiento del sistema es probablemente bajo.
* Nuevas componentes con distinta representación de datos son difícilmente integradas al sistema.
* Modificar el formato de los datos del repositorio implica cambios en todas las componentes.

## Llamada y Retorno: Arquitectura 3 capas

La filosofía de esta arquitectura, se basa en dividir el software en partes o capas, de manera que cada capa sea lo más independiente posible con respecto a las demás. De esta forma en caso de tener que cambiar alguna de las partes, el resto mantengan su funcionalidad y no necesiten ser modificadas.



**Capa de Presentación**

Es la interfaz, la que se encarga de que el sistema interactúe con el usuario y viceversa, muestra el sistema al usuario, le presenta la información y obtiene la información del usuario en un mínimo de proceso. Esta capa se comunica únicamente con la capa intermedia o de negocio.

**Capa de Negocio**

Se denomina capa de negocio o capa de lógica del negocio, porque es aquí donde se establecen todas las reglas que deben cumplirse.

Esta capa se comunica con la de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de acceso a datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él.

**Capa de Datos o Acceso a Datos**

Esta capa es la encargada de almacenar los datos del sistema y de los usuarios. Su función es almacenar y devolver datos a la capa de negocio.