Departamento de Lenguajes y Sistemas Informáticos

BLOQUE I: Arquitectura del Software

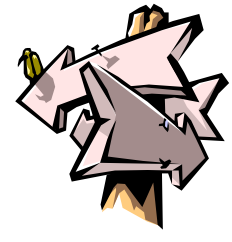
Introducción a la Arquitectura del Software

Tema 2

Arquitectura e Integración de Sistemas Software

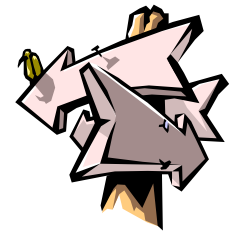
Curso 2012/2013

Índice

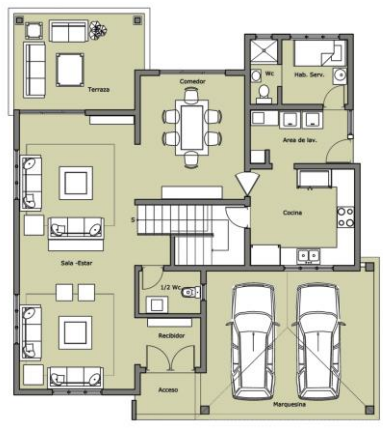


Índice

- **Introducción**
- Arquitectura del software
- Estilos y patrones arquitectónicos
- Artefactos reutilizables
- Principios de diseño
- Resumen
- Bibliografía



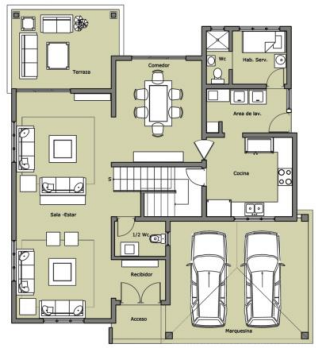
Introducción
Ejemplo de otro dominio



"El diseño de la arquitectura del software es el equivalente del plano de una casa. Éste ilustra la distribución general de las habitaciones, su tamaño, forma y relaciones entre ellas, así como las puertas y ventanas que permiten el movimiento entre los cuartos"

R. S. Pressman

Introducción
Ejemplo de otro dominio

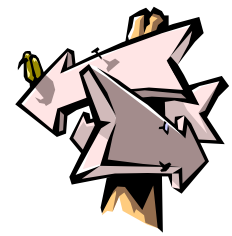


Diseño de la arquitectura

Implementación de la arquitectura

Índice

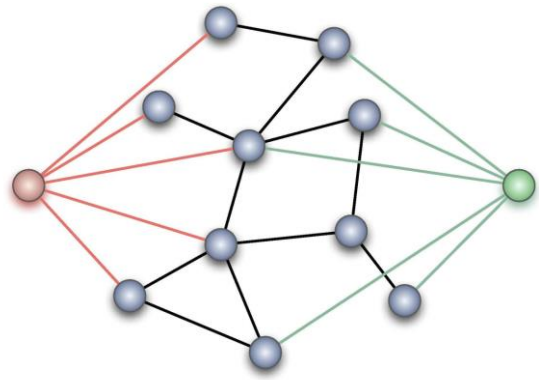
- Introducción
- **Arquitectura del software**
- Estilos y patrones arquitectónicos
- Artefactos reutilizables
- Principios de diseño
- Resumen
- Bibliografía



Arquitectura del software
Definición

"*La **arquitectura del software** de un programa es la estructura o estructuras del sistema, lo que comprende a los componentes del software, sus propiedades externas visibles y las relaciones entre ellos*"

Bass et al.



Arquitectura del software
Definición

"***Software architecture** is the set of design decisions which, if made incorrectly, may cause your project to be cancelled.*"

E. Woods

Arquitectura del software
¿Por qué es importante?

Durante la **gestión del sistema:**

- **Documento** sobre el que poder discutir.
- Aumenta la **precisión** en la **estimación** del coste y tiempo.
- Ayuda a **gestionar la complejidad** (abstracción).

Durante el **desarrollo del sistema:**

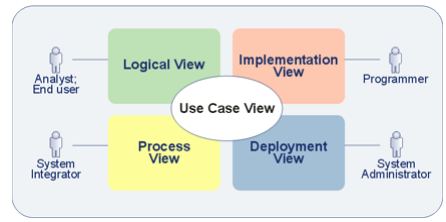
- Es **una excelente vista general** del sistema.
- Proporciona la relación de **puntos de diseño** a tratar.
- Facilita el **desarrollo simultáneo** de componentes.
- Permite **detectar errores** de diseño en fases tempranas.

Arquitectura del software
¿Qué información incluye un diseño arquitectónico?

El diseño de la arquitectura incluye:

- **Estructura: descripción de subsistemas como composición de componentes.**
- **Comportamiento: descripción de la comunicación entre componentes.**
- Cualquier otra información que el arquitecto considere relevante:
  - Protocolos de comunicación, sincronización y acceso a datos.
  - Distribución física.
  - Puntos de variabilidad y extensión.
  - Aspectos de seguridad.
  - …

Arquitectura del software
¿Qué información incluye un diseño arquitectónico?

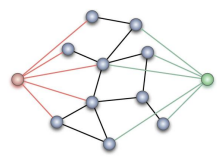


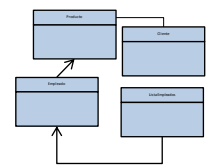
El documento de diseño arquitectónico **suele incluir varias vistas**. Una **vista** representa un aspecto parcial de una arquitectura software. Cada participante en el desarrollo estará interesado en una o varias vistas.

Arquitectura del software
¿Cuándo se diseña la arquitectura?

Es el **primer paso en el diseño** de un sistema, previo al diseño detallado.

Analizar el problema → Diseño arquitectura → Diseño detallado

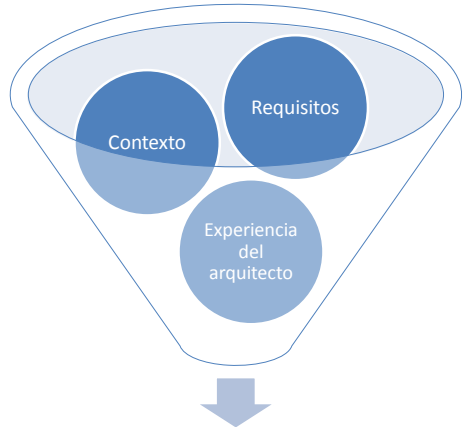




Arquitectura del software
¿Cuándo se diseña la arquitectura?

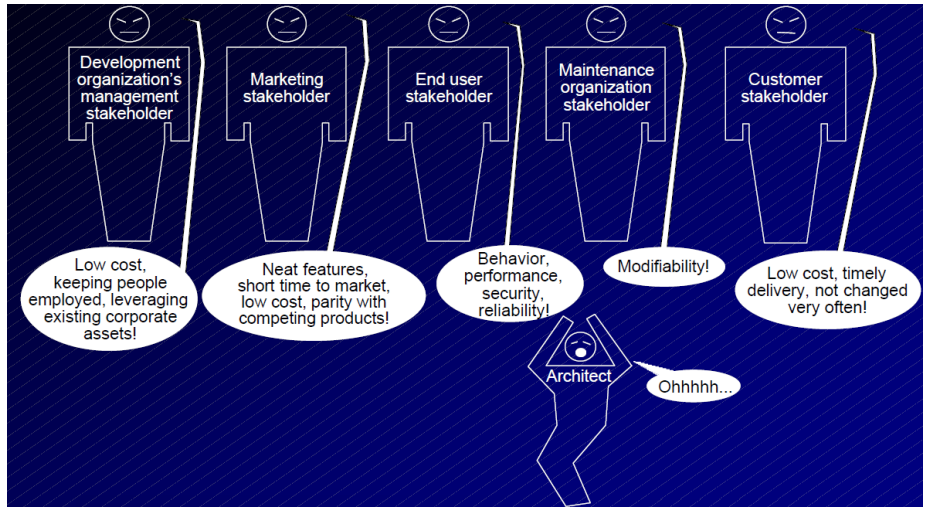
El diseño de la arquitectura podrá evolucionar a lo largo del desarrollo y el tiempo de vida de la aplicación. **El diseño arquitectónico es una actividad continúa.**

Arquitectura del software
¿Qué se tiene en cuenta para el diseño?



Documento de diseño arquitectónico
(Arquitectura Software)

Arquitectura del software
¿Qué se tiene en cuenta para el diseño?- Requisitos



Arquitectura del software
¿Qué se tiene en cuenta para el diseño? - Contexto

Aspectos del negocio:

- Amortizar la infraestructura.
- Mantener bajos costes de instalación.
- Utilizar personal disponible, etc.

Aspectos de la estructura organizacional:

- Promoción de los intereses creados, ej. mantener una BD existente.
- Mantener el método estándar de hacer negocio, etc.

Tendencias actuales:

- Desplegar la aplicación en la nube.
- Interfaz para aplicaciones móviles, etc.

Tecnología disponible: sistema centralizado vs distribuido, desarrollo desde cero vs. uso de servicios externos, etc.

Arquitectura del software
¿Qué se tiene en cuenta para el diseño? - Arquitecto

- Los arquitectos desarrollan su modo de pensar a partir de experiencias anteriores.
  - Experiencias anteriores buenas darán lugar a replicar esos diseños.
  - Las experiencias anteriores malas se evitarán en el nuevo diseño.

Arquitectura del software
¿Quién diseña y gestiona la arquitectura?

El **arquitecto software**…

- Es un líder técnico.
- Puede ser un equipo. Posibles roles: arquitecto jefe (necesario), arquitecto de aplicaciones, arquitecto de infraestructuras, arquitecto de datos, etc.
- Tiene conocimientos tecnológicos y habilidades de programación.
- Tiene habilidades para el diseño.
- Conoce el dominio del negocio.
- Es consciente de las políticas organizacionales.
- Es un buen comunicador.
- Toma decisiones.
- Es un negociador.

Arquitectura del software
¿Cómo se modela?

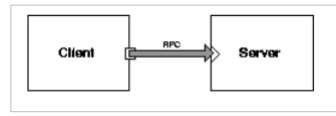
La arquitectura de un sistema software puede modelarse mediante:

- Lenguajes de Descripción de Arquitectura (ADLs)
- Diagramas UML
- Diagramas de bloque de alto nivel

Arquitectura del software
¿Cómo se modela? - ADLs

Lenguajes de descripción de arquitectura (ADLs). Lenguajes textuales formales ideados para describir una arquitectura software en términos de componentes y conectores.

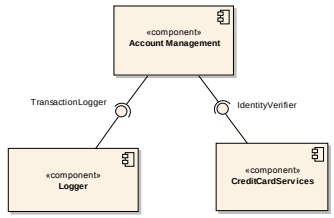
```
System simple_cs = {
    Component client = { Port send-request; };
    Component server = { Port receive-request; };
    Connector rpc = { Roels { caller, callee}};
    Attachments {
        client.send-request to rpc.caller;
        server.receive-request to rpc.callee;
    }
}
```

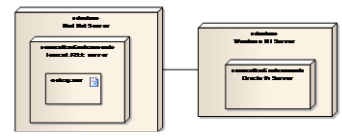


Arquitectura del software
¿Cómo se modela? – Diagramas UML

Diagramas UML

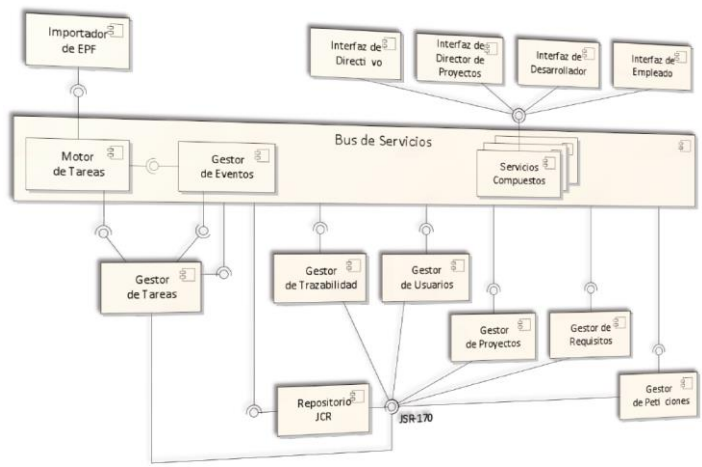
- **Descripción de aspectos estructurales (estático):**
  - Diagrama de componentes, diagrama de despliegue, etc.
- **Descripción de comportamiento (dinámico)**
  - Diagrama de actividad, diagrama de secuencia etc.





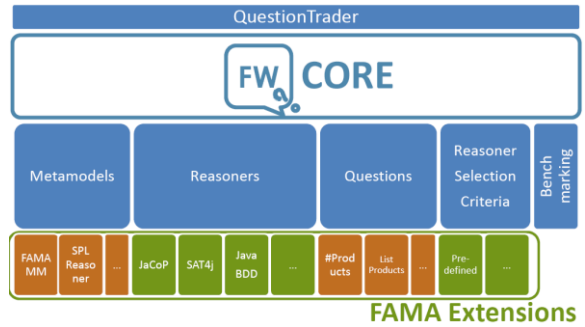
Arquitectura del software
Ejemplo

Diagrama de componentes (Proyecto Alcuza)



Arquitectura del software
¿Cómo se modela? – Diagramas de alto nivel

- Describen la arquitectura con una **alto nivel de abstracción**. Suele ser el primer paso antes de un diseño más detallado (ej. con diagramas de componentes)



Arquitectura del software
Ejemplo: Arquitectura de Tuenti

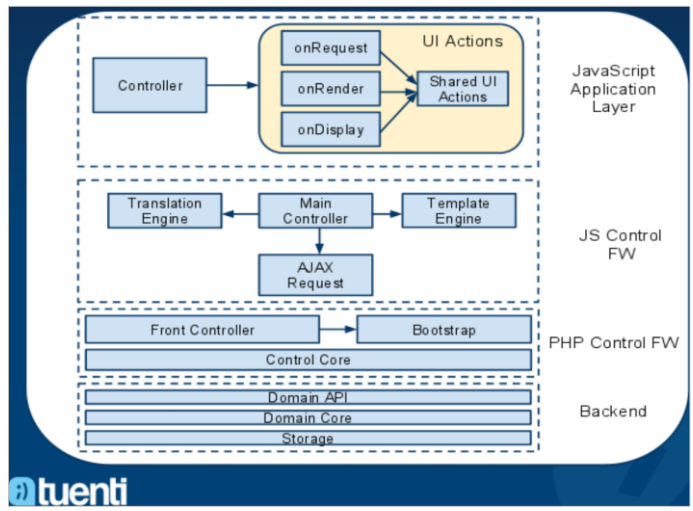


Imagen tomada de www.slideshare.net/gurbani/tuenti-architecture

Arquitectura del software
Ejemplo: Arquitectura de Facebook

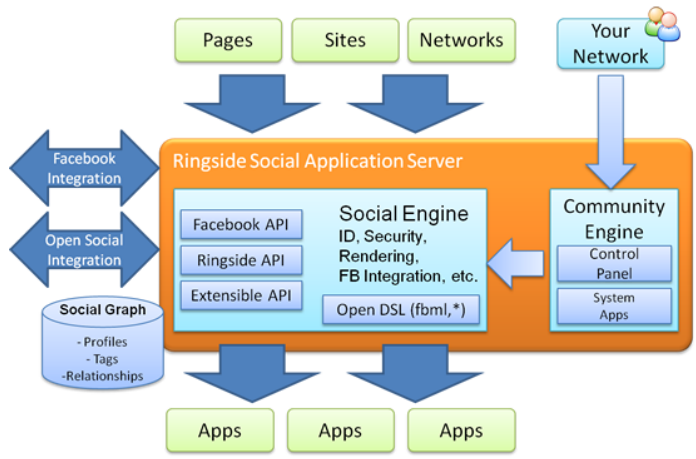
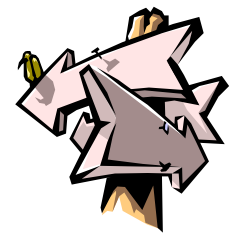


Imagen tomada de http://www.fmderana.lk/index.php?route=entertainment/kik_news&nid=196

Índice

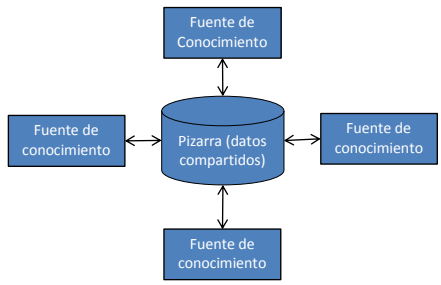
- Introducción
- Arquitectura del software
- **Estilos y patrones arquitectónicos**
- Artefactos reutilizables
- Principios de diseño
- Resumen
- Bibliografía



Arquitectura del software
Estilos y patrones arquitectónicos

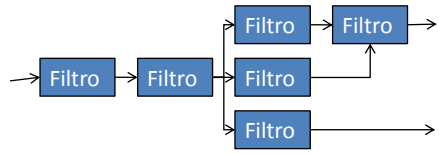
- Un **diseño arquitectónico** se refiere a la arquitectura de un sistema concreto.
- Un **estilo arquitectónico** establece las restricciones sobre la arquitectura de una familia de diseños arquitectónicos. Algunos ejemplos: centrado en datos, flujo de datos, llamar y regresar, capas.
- Un **patrón arquitectónico** es una solución general a un problema común del diseño arquitectónico. Menor alcance que los estilos arquitectónicos.

Arquitectura del software
Estilos arquitectónicos: Centrado en datos (Blackboard)



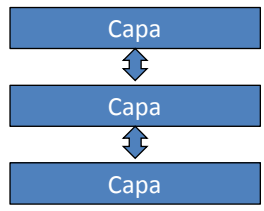
- El centro de la arquitectura es una "**pizarra**" y otros componentes tienen acceso a ella para actualizar, agregar, eliminar o consultar sus datos.
- Facilita la integración pues los componentes son independientes.
- Se puede pasar datos entre componentes a través del almacén de datos.

Arquitectura del software
Estilos arquitectónicos: Tuberías y filtros (Pipes and filters)



- Se aplica cuando los datos de entrada se han de transformar en datos de salida mediante una serie de operaciones.
- Los componentes (**filtros**) van transmitiendo datos al siguiente por medio de **tuberías**.
- Los filtros no necesitan saber el funcionamiento de los vecinos. Sólo se preocupan de su entrada y su salida.
- Si hay una sola línea de transformaciones se denomina procesamiento por lotes secuencial (pipeline).

Arquitectura del software
Estilos arquitectónicos: Capas (Layers)



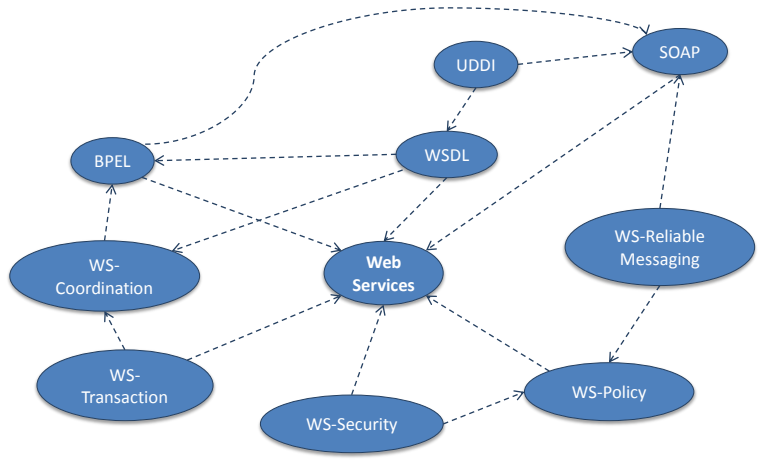
- Se definen distintas capas en la aplicación de manera que sólo se comunican entre si las capas adyacentes.
- Favorece el bajo acoplamiento.
- Este es el estilo arquitectónico empleado en las aplicaciones web convencionales.

Arquitectura del software
Estilos arquitectónicos: Arquitectura Orientada a Servicios (SOA)

Las **arquitecturas orientadas a servicios (SOA)** son un estilo arquitectónico basado fundamentalmente en el uso de servicios web.

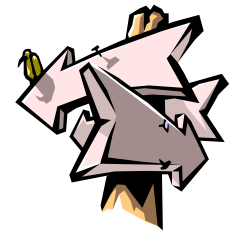
Arquitectura del software
Estilos arquitectónicos: Arquitectura Orientada a Servicios (SOA)

Uso de estándares.



Índice

- Introducción
- Arquitectura del software
- Estilos y patrones arquitectónicos
- **Artefactos reutilizables**
- Principios de diseño
- Resumen
- Bibliografía



Artefactos reutilizables

Un buen arquitecto no reinventa la rueda

Artefactos reutilizables
Componentes

- **Un componente es una unidad modular con interfaces bien definidas, que es reemplazable dentro del contexto (OMG).**
- Los componentes definen su comportamiento en términos de interfaces proporcionadas y requeridas.
- Un componente debe poder ser reemplazable por otro que cumpla con las interfaces declaradas.
- Las características fundamentales que debe cumplir son:
  - Ser reutilizable.
  - Ser intercambiable.
  - Poseer interfaces definidas.
  - Ser altamente cohesivos.

Artefactos reutilizables
Servicios Web

Un servicio web ofrece una interfaz de programación (no de usuario) de una determinada funcionalidad (servicio) accesible a través de Internet y basada en estándares W3C.

Permiten intercambiar información entre aplicaciones software desarrolladas en lenguajes de programación diferentes y ejecutadas sobre distintas plataformas. **Son la base para la integración Web.**

Ventajas:
- Facilita la interoperabilidad entre sistemas diversos.
- Uso de protocolos abiertos.

Inconvenientes:
- Rendimiento.

Artefactos reutilizables
Librerías

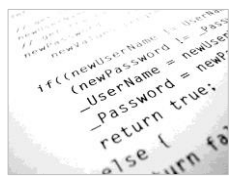
- **Una librería proporciona código reutilizable que puede ser empleado por los desarrolladores de aplicaciones.**
- Ejemplo: Librería de funciones matemáticas Math.lib
- Las librerías pueden incorporarse al programa en tiempo de compilación (ej. lib,jar) o en tiempo de ejecución (ej. DLLs).
- El desarrollo de librerías puede ser una buena idea para facilitar el uso de nuestras aplicaciones por parte de terceros. Ej. Amazon SDK para Java.

Artefactos reutilizables
Frameworks

- **Un framework es un conjunto de clases parcialmente funcional (no es una aplicación) para un dominio de aplicación.**
- Les falta aquello que es propio de la aplicación.
- Los frameworks suele implementar distintos patrones y estilos arquitectónicos.
- El uso de frameworks va a determinar en gran medida la arquitectura del sistema.

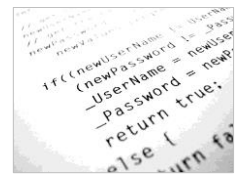
Artefactos reutilizables
Librerías vs. frameworks

Librería



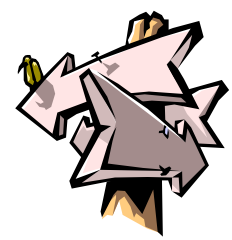
Código de nuestra aplicación

Framework



Índice

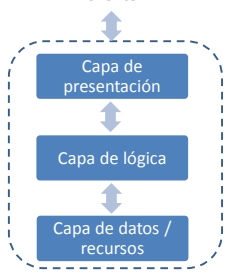
- Introducción
- Arquitectura del software
- Estilos y patrones arquitectónicos
- Artefactos reutilizables
- **Principios de diseño**
- Resumen
- Bibliografía



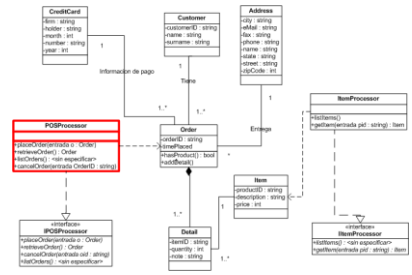
Principios de diseño

Nociones clave a tener en cuenta para el diseño efectivo de sistemas software.

Abstracción. Omitir detalles no relevantes.



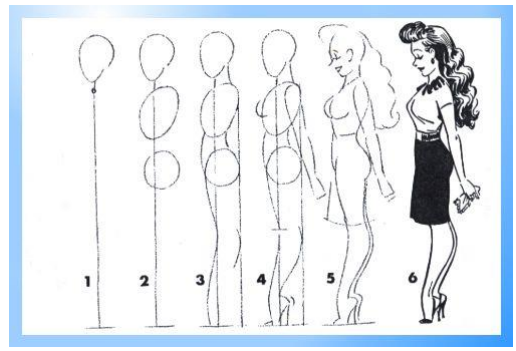
Alto nivel de abstracción.



Bajo nivel de abstracción.

Principios de diseño

- Para el diseño de la arquitectura se recomienda comenzar con un alto grado de abstracción y refinar sucesivamente hasta llegar al nivel de componente.



Principios de diseño

Descomposición. Dividir los problemas en problemas más pequeños. (Divide y vencerás)

Principios de diseño

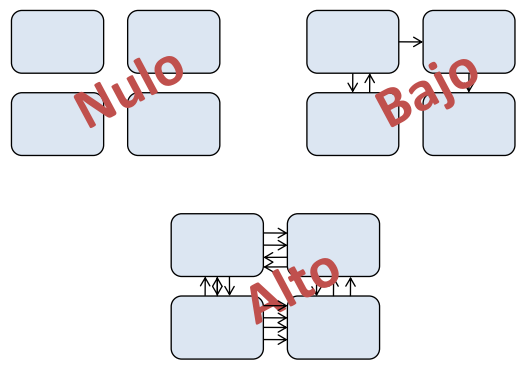
Cohesión. Es un indicador cualitativo del grado en el que un módulo se centra en hacer una sola cosa. **La cohesión de un diseño debería ser alta.**

"Un elemento es altamente cohesivo si todos sus elementos trabajan juntos para proporcionar algún comportamiento bien delimitado"

Grady Booch

Principios de diseño

Acoplamiento. Es un indicador cualitativo del grado en el que un módulo está conectado con otros y el mundo exterior. **El acoplamiento debe ser bajo.**



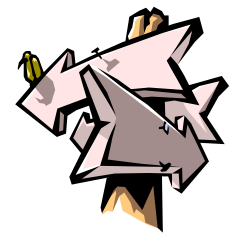
Principios de diseño

Variaciones protegidas. Intentar ocultar/proteger de los cambios al resto del sistema.

Objetivo: Lograr que los cambios involucren la menor cantidad de código posible y estén lo más acotados posible.

Índice

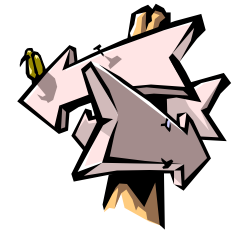
- Introducción
- Arquitectura del software
- Estilos y patrones arquitectónicos
- Artefactos reutilizables
- Principios de diseño
- **Resumen**
- Bibliografía



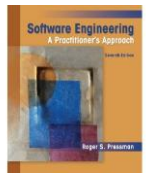
Resumen
¿Qué hemos aprendido?

- La arquitectura software…
  - Define la estructura y el comportamiento del software.
  - Puede tener múltiples vistas.
  - Es el primer paso en la fase de diseño. Es un actividad continua.
  - Se diseña a partir de los requisitos, el contexto y la experiencia del arquitecto.
  - Se modela mediante ADLs, diagramas UML y/o diagramas de bloque.
- Artefactos reutilizables: componentes, librerías, frameworks…
- Estilos vs. patrones arquitectónicos.
- Estilos arquitectónicos: centrado en datos, tuberías y filtros, capas, arquitecturas orientadas a servicios.
- Principios de diseño a tener en cuenta: cohesión, acoplamiento…

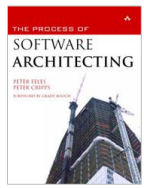
Índice



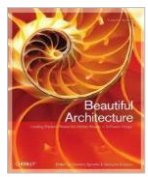
Bibliografía



Pressman R. *Software Engineering: A Practitioner's Approach.* McGraw-Hill. 2009 (7th edition)



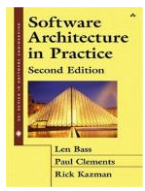
The Process of Software Architecting, Peter Eeles y Peter Cripps. Addison-Wesley, 2009



Beautiful Architecture: Leading Thinkers Reveal the Hidden Beauty in Software Design, Diomidis Spinellis, Georgios Gousios. O'Reilly Media, Inc., 2009

Bibliografía

Buschmann F. *et al .Pattern-Oriented Software Architecture.* John Wiley & Sons. 1996



Bass L. *et al*. *Software Architecture in Practice* Addison-Wesley Professional. 2003

Alonso G. *et al. Web Services Concepts, Architectures and Applications*. Springer. 2004

Disclaimer and Terms of Use

All material displayed on this presentation is for teaching and personal use only.

Many of the images that have been used in the presentation are Royalty Free images taken from http://www.everystockphoto.com/. Other images have been sourced directly from the Public domain, from where in most cases it is unclear whether copyright has been explicitly claimed. Our intention is not to infringe any artist's copyright, whether written or visual. We do not claim ownership of any image that has been freely obtained from the public domain. In the event that we have freely obtained an image or quotation that has been placed in the public domain and in doing so have inadvertently used a copyrighted image without the copyright holder's express permission we ask that the copyright holder writes to us directly, upon which we will contact the copyright holder to request full written permission to use the quote or images.