
Universidad
Tecmilenio®

Universidad Tecmilenio

Ingeniería de Software

Conceptos de diseño

Semana 6

Bienestar-mindfulness

Te invito a realizar la siguiente actividad de bienestar-mindfulness antes de comenzar a revisar el tema:

https://youtu.be/16yTNMiA5Ks

Introducción

El método de diseño arquitectónico basado en la funcionalidad se centra principalmente en los requerimientos funcionales documentados en la ingeniería de requerimientos.

Explicación

Los pasos para describir un diseño arquitectónico basado en funcionalidad son:

- Definir contexto
- Identificar arquetipos y sus relaciones
- Describir instancias del sistema

Explicación

El diseño estructurado es un método que transforma el análisis elaborado en un Diagrama de Flujo de Datos (DFD) a la arquitectura de software.

Los pasos del método son:

Revisión del modelo del sistema fundamental	Revisar y mejorar los diagramas de flujos para el software	Determinar si el DFD tiene características
Aísla el centro de transformación	Rediseño del primer nivel	Rediseño del segundo nivel

Explicación

El refinamiento del diseño debe perseguir el menor número de componentes, que sea consistente con la modularidad efectiva y la estructura de datos menos compleja que cumpla de modo adecuado con los requerimientos de información.

Ejercicio

Desarrolla un diagrama que muestre el diseño estructurado de un sistema de vigilancia de velocidad en una avenida concurrida de la ciudad.

Cierre

El diseño de la arquitectura de un software es un proceso iterativo en el que se prueban diferentes estructuras hasta encontrar la que mejor resuelva los requerimientos del usuario.

El arquitecto de sistemas debe tener en mente la necesidad de establecer una estructura lo más simple posible, que pueda posteriormente modificarse o adaptase a nuevas necesidades.

Bibliografía

- Bosch, J. (2000). Design & use of software architecture. EE. UU: Addison-Wesley.
- Pressman, R. (2010). Ingeniería de Software. Un enfoque práctico (7ª ed.). México: McGraw-Hill.

Universidad Tecmilenio

Ingeniería de Software

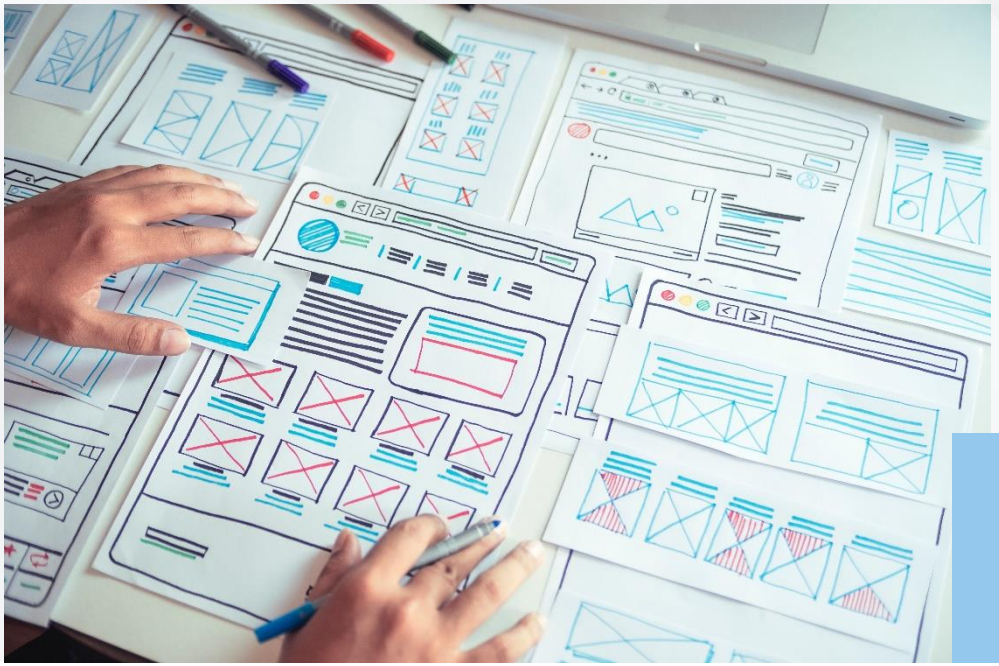
Fundamentos de los patrones

Semana 6

Introducción

Los patrones en el diseño del software reutilizan una estructura que ha sido exitosa al usarse en diferentes escenarios, sin la necesidad de tener que pasar por los mismos errores cometidos por otros.

Estos patrones deben ser considerados dentro de la etapa del diseño porque tienen un impacto en la arquitectura general del software.



Explicación

El patrón es una descripción del problema y la esencia de su solución, de modo que la solución puede reutilizarse en diferentes configuraciones.

Explicación

Los pasos para que un diseñador de sistemas piense en patrones son:

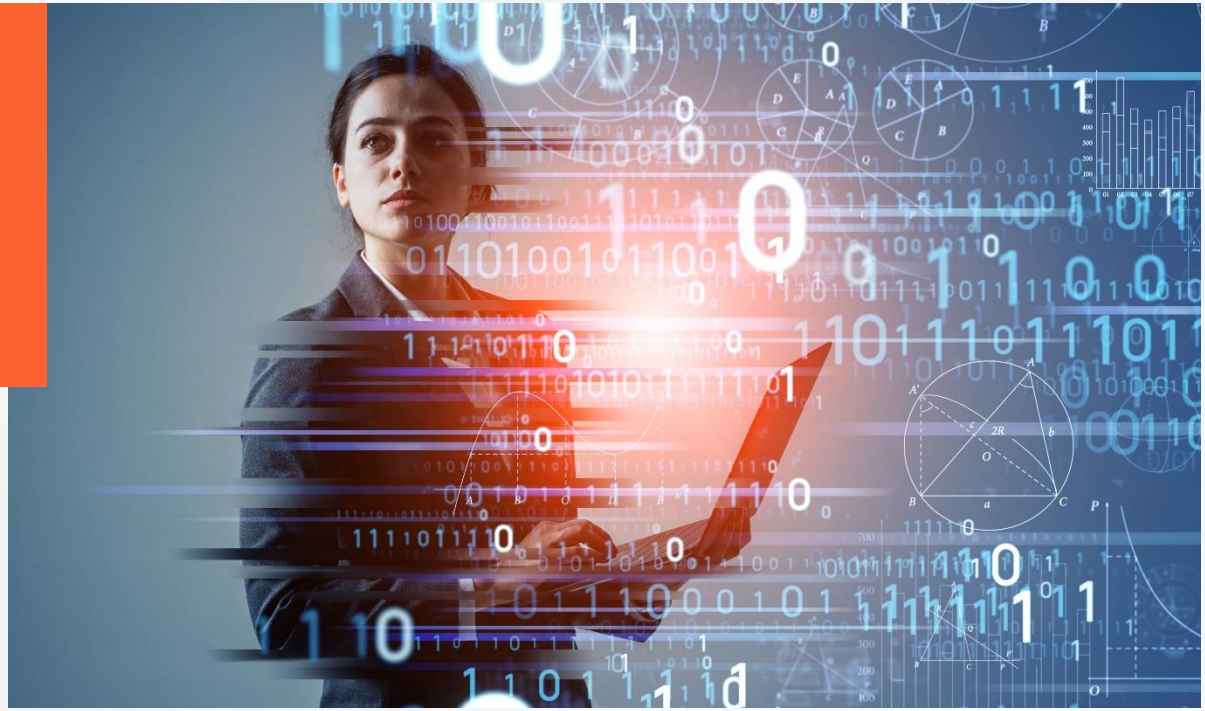
- Entender el contexto
- Identificar patrones
- Diseñar patrones alto y bajo nivel
- Refinar
- Mejorar el diseño

Explicación

La estructura es un esqueleto de una infraestructura específica que puede complementar un patrón de diseño, permitiendo así describir de manera más completa el diseño del software.

Ejercicio

Enlista cinco patrones de diseño utilizados en aplicaciones web.



Cierre

Los patrones de diseño ofrecen un camino para solucionar problemas que enfrenta el diseño del software.

Los patrones son una fuente de información muy valiosa para explicar la configuración, uso e implementación de tus diseños al grupo técnico de desarrollo, además de que pueden ser una herramienta de entrenamiento muy eficaz para nuevos integrantes del equipo.

Bibliografía

- Gamma, E., Helm, R., Johnson R. y Vlissides J. (1995). Design Patterns, Elements of Reuseable, Object-Oriented Software. EE.UU: Addison – Wesley.
- Hanmer, R. (2013). Pattern-Oriented Software Architecture For Dummies. EE. UU: Wiley.
- Pressman, R. (2010). Ingeniería de Software. Un enfoque práctico.(7a. ed.). México: McGraw Hill.
- Shalloway, A y Trott, J. (2005). Design Patterns Explained. (2a ed.). EE. UU.: Addison – Wesley.
- Sommerville, I. (2011). Ingeniería de Software. (9ª ed.). México: Pearson.
- VirtuFropple. (2012). Patrones de diseño. Recuperado de http://patronesdediseno.net16.net/

Universidad Tecmilenio

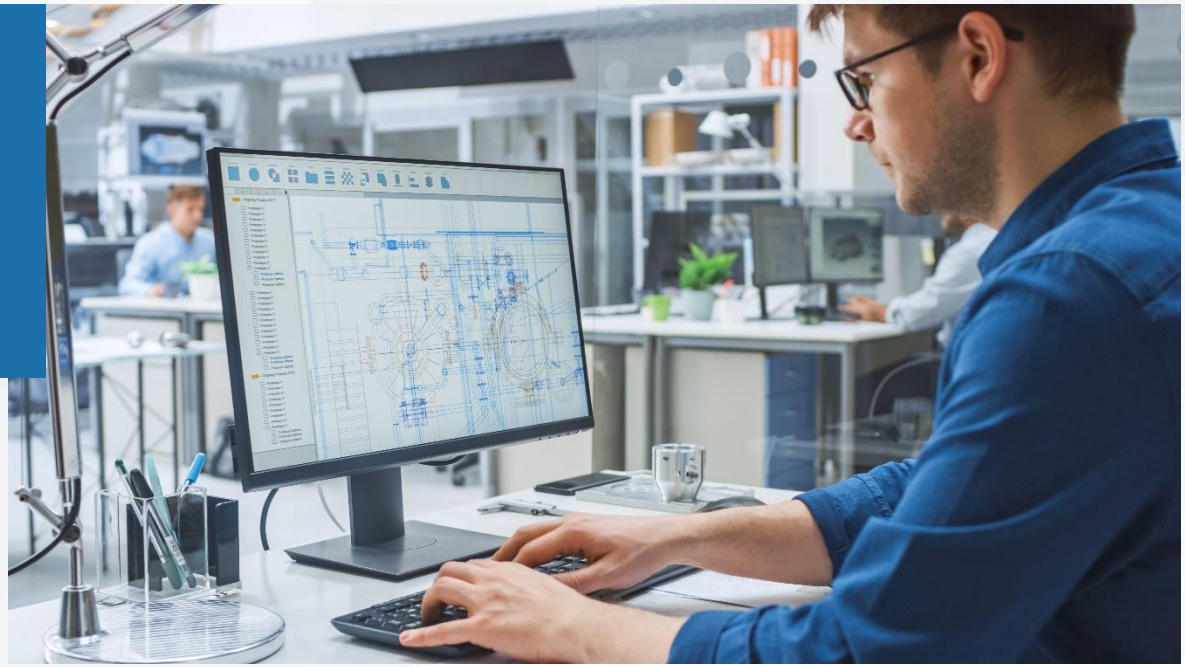
Ingeniería de Software

Estructuras genéricas del software

Semana 6

Introducción

Los patrones arquitectónicos permiten establecer una estructura general del software que ha sido empleada en diseños de aplicaciones de forma exitosa.

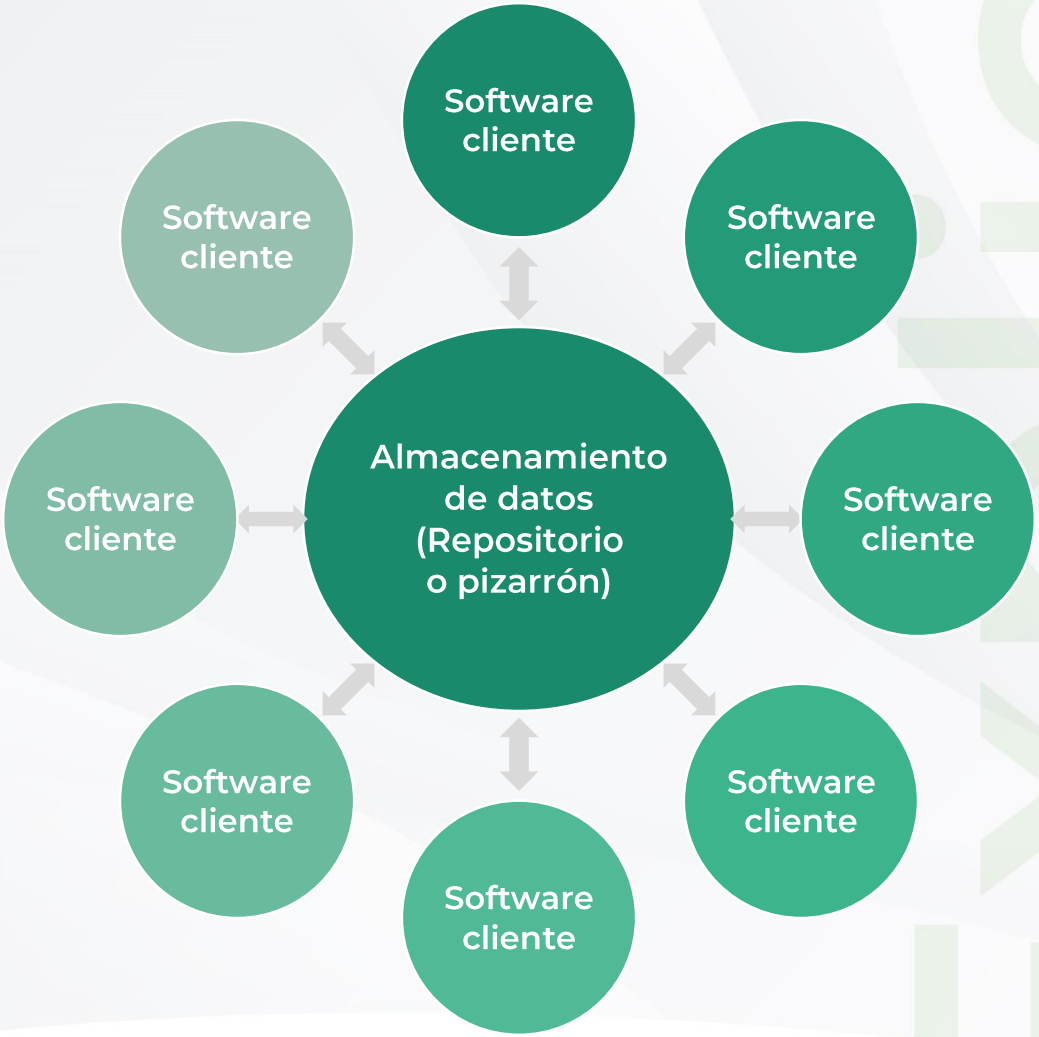


Explicación

La arquitectura en capas permite separar en componentes independientes las características del software, ofreciendo la flexibilidad de realizar cambios de manera incremental sin afectar elementos del sistema que pertenecen a otras capas.

Explicación

La arquitectura de repositorio permite compartir y manipular grandes cantidades de datos a través de un repositorio central de información, un componente que administra el acceso a la información y un conector que lee y escribe los datos.



Architectura centrada en datos.
Diagrama tomado del libro Pressman, R. (2010). Ingeniería de Software. Un enfoque práctico. (7ª. Ed.). México: McGraw Hill. Fig. 9.1.
Sólo para fines educativos.

Explicación

La arquitectura MVC separa los elementos del sistema en componentes independientes, cada uno con una función específica, razón por la cual es muy parecida al patrón arquitectónico de capas.

Modelo

- Encapsula el estado de la aplicación.

Vista

- Interpreta el modelo.
- Solicita actualizaciones del modelo.
- Envía eventos de usuario al controlador.

Controlador

- Mapea acciones del usuario para modelar actualización.

Ejercicio

Enlista cinco ventajas del modelo vista controlador.

Cierre

Los patrones arquitectónicos genéricos que viste en este tema son soluciones que han sido utilizadas con éxito en varias situaciones.

Cada arquitecto de software puede incluir ciertos ajustes al patrón que son importantes para el desarrollo del sistema y dan una mejor respuesta al software por desarrollarse.

Bibliografía

- Bass, L., Clements, P., y Kazman, R. (2012). Software Architecture in Practice. EE. UU: Addison - Wesley Professional.
- Hanmer, R. (2013). Pattern-Oriented Software Architecture For Dummies. EE. UU: Wiley
- Pressman, R. (2010). Ingeniería de Software. Un enfoque práctico. (7a. ed.). México: McGraw-Hill.
- Sommerville, I. (2011). Ingeniería de Software (9ª ed.). México: Pearson.

Universidad Tecmilenio

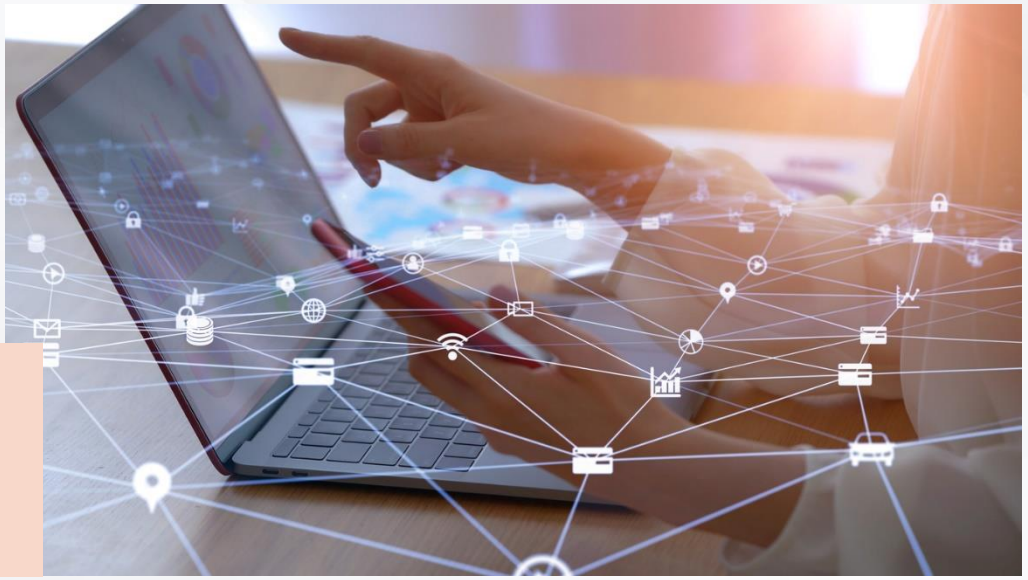
Ingeniería de Software

Arquitectura de sistemas distribuidos

Semana 6

Introducción

El sistema distribuido es una configuración en la que el hardware se encuentra interconectado, haciendo que el lugar físico y el tipo de equipo sean recursos asequibles y a la vez transparentes para el usuario.



Explicación

Las ventajas de los sistemas que tienen un enfoque distribuido son las siguientes:

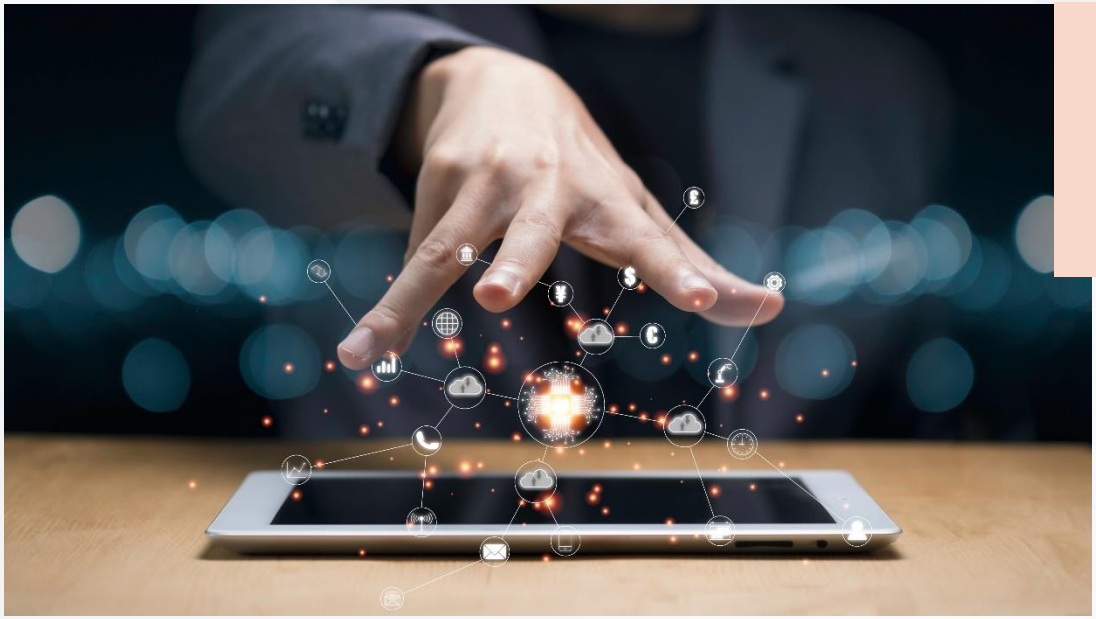
- Compartición de recursos
- Apertura
- Concurrencia
- Escalabilidad
- Tolerancia a fallas

Explicación

La arquitectura cliente-servidor es la más extendida dentro de las arquitecturas de los sistemas distribuidos, ya que permiten administrar las funcionalidades y los recursos en dos elementos fundamentales: un cliente y un servidor, mientras que para el usuario, el acceso a aplicaciones, bases de datos y servicios es transparente.

Explicación

En la arquitectura Peer to Peer el procesamiento puede realizarse en cada uno de los clientes, aprovechando así su poder de computación y almacenamiento.



Ejercicio

Enlista cinco ventajas de la arquitectura cliente-servidor y cinco ventajas de la arquitectura Peer to Peer.

Cierre

Los sistemas distribuidos son implementados a través de las arquitecturas cliente-servidor, peer to peer y orientados al servicio.

Cada una de ellas ofrece la posibilidad de ubicar diferentes servicios en distintos lugares, compartir información con entidades ajenas a la organización y aprovechar recursos de cómputo.

Bibliografía

- Bass, L., Clements, P. y Kazman, R. (2012). Software Architecture in Practice. EE. UU: Addison - Wesley Professional.
- Pohl, K. y Rupp, Chris. (2011). Requirements Engineering Fundamentals. EE. UU: Rocky Nook.
- Sommerville, I. (2011). Ingeniería de Software (9ª ed.). México: Pearson.