**eArquitectura de Software**

**Profesor:** Henry Coral

**Temas:**

* **Conceptos de Arquitectura de Software**
  + **Patrones Arquitecturales: Formas de diseñar sistemas**
    - **Cliente-Servidor**
    - **MVC**
      * **Navegación**
      * **Presentación**
      * **Procesamiento de Datos**
    - **N-Capas**
      * **Separación de cada una de las capas lógicas**
        + **Datos**
        + **Acceso a Datos**
        + **Servicios / Reglas de Negocios**
        + **Presentación**

**JSF**

**Vistas - XHTML**

**Controlador – Faces Servlet (Navegación entre vistas) - Beans**

**Modelo – Session Bean**

* + - **Se puede mezclar diversos patrones**
    - **SOA – Service Oriented Architecture**
      * **SOAP**
      * **REST**
    - **Microservices architecture**
      * **Diferentes stacks de tecnología con diferentes repositorios de datos con un solo front end que se conecta con todo por rest.**
  + **Patrones de Diseño: Recetas probadas para resolver problemas comunes**
    - **Estructurales**
    - **Comportamiento**
    - **Creación**

**SOA 🡪**

**Arquitectura de Software**

Tiene sus componentes, relaciones entre ellos, patrones de composición y restricciones

Un diagrama de arquitectura con tiene todos los elementos del sistema. Base de datos corre del 1 al 49151.

JSF 2.2

PrimeFaces 7

PAG.

Xhtml

Servlets

OCI-Oracle

Protocolo

Base de Datos

Oracle 12c

Replicación

Linux RedHat 9

JDBC

Protocolo

Base de Datos

Oracle 12c

Servicios

DAO

Clases Persistencia

JPA 2.x

EJB 3.x

La arquitectura de software se ocupa de:

Diseño preliminar de alto nivel

Organizaicion de alto nivel del sistema, descripción y análisis de propiedades de estructura y control, los protocolos e comunicación y sincronización, distribución física y componentes

Estilo o comninacion de estilos

Requerimientos no funcionales

Otros aspectos relacionados co el desarrollo del sistema y su evolución y adaptación al cambio.

Composición, reconfiguración, reutilización, escalabilidad, mantenibilidad, etc.

Esencial para el éxito o fracaso de un proyecto.

Lo que no hace

Diseño detallado

Diseño de algoritmos

Diseño de estructuras de datos

Diseño de software con UML

Metodología

Arquitectura vs Diseño

Abstracción es diferente, arq es alto y dis es bajo

Entregables para arq son planear subsistemas, interfaces con sistemas externos, servicios horizontales, frameworks, componentes reutilizados, prototipo arquitectónico. El dis, da diseño detallado de componentes, especificación de codificación.

Areas de Enfoque para uno son tecnologías, requerimientos no funcionales y manejo de riesos, para el otro, requerimientos funcionales.

**Estilos Arquitectónicos**

Indican:

* Los tipos de componentes y conectores involucrados
* Patrones y restricciones de interconexión o composición entre ellos: Invariantes del estilo

Asociados a cada estilo hay una serie de propiedades que lo caracterizan.

Pocos estilos what?

Tipos:

Estilos de Flujos de Datos

* Tuberías y Filtros: Como una red neuronal, un filtro tiene una entrada o varias y retorno una salida o varias salidas y así se pasa esos resultados a otros filtros.
  + Los filtros deben ser independientes
  + Los filtros realizan la labor independientemente del flujo de entrada.
  + Especializaciones
    - Pipelines
    - Bounded pipes
    - Typed pipes
  + Ventajas:
    - Permite entender el sistemas en forma de agrupación de componentes
    - soporta reutilización
    - Facilidad de mantenimiento y mejora
    - Facilidad de diagnostico
    - Soporta ejecución concurrente
  + Desventajas
    - No aconsejado cuando se necesita interactividad.
    - Problemas de performance ya que los datos se transmiten en forma completa entre filtros.
  + Se usa para procesos de fin de día de los bancos, reportes y similares de bloques grandes de datos.

Estilos Centrados en Datos

* Arquitectura de Pizzara o Repositorio: Hay una pizzara que recibe información de indicadores de funcionamiento de diferentes elementos. Permite el monitoreo de elementos. Usado en hidroeléctricas, centrales nucleares, un auto, etc.
  + Una escutruca central de datos
  + Componentes independientes
  + Ventajas
    - Integración de agentes
    - Adecuado para la resolución de problemas no deterministas
    - Se puede resumir el estado de conocimiento en cada momento del proceso
  + Desventajas
    - Estructura de datos es común para todos los agentes
    - Problemas de carga a la hora de chequear y vigilar el estado de la pizarra.

Estilo de Llamada y Retorno

* Arquitectura en Capas
  + En capas, donde cada capa da servicios a la superior y servido por la capa inferior
  + Ventajas
    - Descomposición del problema
    - Mejora fácil
  + Desventajas
    - No todos los sistemas pueden ser estructurado en capas
* Innvocacion basada en eventos: Un componente genera un evento a base de algo que paso
* Ventajas:
  + Simple
  + Modular
  + Puede mejorar eficiencia
* Desventajas
  + Posible Desborde
  + Imprevision de Escalabilidad
  + Pobre comprensibilidad
  + No garantía que se responda al evento

Estilosd de Código Movil (Virtual MACHINE)

Estilos Heterogeneos

Estilos Peer to Peer

**Relaciones entre estilos y patrones**

**Patrones**

Solución a un problema constante en algún campo, para ser aplicado constantemente por que siempre funciona.

**Patrones de Arquitectura**

Esquema organizativos

* También llamados arquetipos
* Expresa un esquemas de organización estructural esencial para un sistema de software, que costa de subsistemas, sus responsabilidades e interrelaciones.
* Mayor nivel de abstracción.
* MVC: Modelo, Vista y Controlador
  + Modelo: Lógica de Negocio
  + Controlador: Flujo de Aplicación
  + Vista: Presentación

**Parones de Diseño**

Bajo nivel

**Frameworks**

Frameworks para diferentes propósitos aplicados para diferentes capas de arquitectura.

Trabajar sin un frameworks vuelve mas complicado el flujo de trabajo ya que estos frameworks son con herramientas para un trabajo específico.

Estructura conceptual y tecnológica de soporte definida, con artefactos de software concretos, mediante el cual otro proyecto puede ser organizado y desarrollado.

* Incluyen usualmente: patrones de diseño, soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Un framewok se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las ultimas piezas para construir una aplicación concreta.

Objetivos:

* Acelerar el proceso de desarrollo
* Adsa
* Das
* D

Ventajas:

* No es necesita planea una estructura global de aplicación
* Facilita la colaboración. Pues al definir y estandarise se ahorra tiempo y trabajo en dev colaborativo
* ES ACCESIBLE
* Menos costos

Frameworks White Box

* La clases y el código de la plaicacion se pueden introducir por herencia o composición.
* Se debe conocer muy bien el framework para usarlo.
* El uso principal se lo hace por la herencia.

Frameworks Caja Negra

* Para utilizar los frameworks no es necesario saber como están construidos
* El dev solo debe configurar el funcionamiento del frameworks de acuerdo a sus necesidades sin tener que extender nada.
* Su uso es configurar correctamente el framework para las necesidades dadas.

**SERVICIOS WEB**

EXAMEN DE PRIMER PARCIAL ES MANEJAR SOAP y talvez REST.

**INTEGRACIÓN DE SISTEMAS**

**LA WEB**

Nació a finales de los 80

Tim Bernes Lee

La world wide web usa internet para funcionar

La web es una aplicación del internet

La web tiene los siguientes componentes:

* Servidor web
* Protocolo HTTP
* Lenguaje HTML
* Navegador

Internet: red conformada por diferentes redes heterogéneas que utiliza el protocolo TCP/IP

Servidor we es una app software que entiende y trabaja el protocolo HTTP. Hace uso del protocolo TCP e IP y manejar comunicación por sockets.

HTTP es un protocolo stateless, es decir que no es necesario que exista un usuario, o una sesión para la comunicación que este almacenada como estado en el servidor. Por lo tanto, no se mantiene una conexión continua por un puerto especifico.

El protocolo solo envia texto

Tiene as siguientes peticiones: GET/POST/PUT/DELETE

Codigo de respuesta:

2xx: Respuestas correctas

3xx: Redirección

4xx: Errores causados por el cliente

5xx: Errores causados por el servidor

HTML

NAVEGADOR

**INTEGRACIÓN DE SISTEMAS**

Importante para las grandes y medianas empresas

Existen diversas frmas de integración

La integración puede ser en tiempo real o en no tiempo real.

Existen limitaciones tecnológicas para la integración a base de las tecnologías y herramientas usadas en los sistemas.

Se puede integrar:

Integración a nivel de datos:

Integración de procesos:

Independencia de proveedor: Ejecutar reglas de negocio y/o combinarlas, evitando la reimplementación.

Fachada común: Un sistema de gestión puede actuar de front end y detrás estarían todos los otros sistemas comunicándose con el front end.

Los problemas son:

Construir interfaces es un Arte:

Las soluciones deben ser negociadas. La falta de consenso en los diseños de las interfaces tiende a acarrear un esfuerzo excesivo.

Falta de detalle:

Información que al principio se ve poco importante, pero a la final terminando ser importantes.

Falta de experiencia en Integración de Sistemas:

La Integración de Sistemas requiere conocimiento de muchas problemáticas y aspectos técnicos.

TECNOLOGIAS DE INTEGRACIÓN

La forma mas básica es:

Archivos Planos: para sistemas heterogéneos, plataformas diferentes

DBLink (Link entre dos bases de datos del mismo tipo)

SOCKETS : Por ejemplo un switch de telefonía celular: heterogeneo

CORBA (Common Object Request Broker Architecture): Heterogeneas

RMI (Remote Method Invocation) es de JAVA: Homogenea

DCOM RMI Para Microsoft: Homogenea

RPC (Remote Procedure Call): Heterogeneas

SERVICIOS REST: Heterogeneas

SERVICIOS SOAP: Heterogeneas

SERVIDORES DE COLAS: Heterogéneas

**SERVICIOS WEB**

Es una tecnología que usa un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.

Aplicaciones de todo tipo pueden intercambiar datos.

La interoperabilidad se logra con estándares abiertos.

Es la integración de sistemas heterogéneos en tiempo real por defecto.

Los servicios web son basados en estándares diferentes de las plataformas.

Los servicios web utilizan el protocolo HTTP para la invocación de operaciones.

Lo que en realidad se invoca son métodos del servicio web (SOAP)

Estándares en servicios web:

XML, SOAP, WSDL, UDDI, WS – SECURITY

REST

XML (eXtended Markup Language)

XML es un subconjunto de SGML (Standard Generalized Markup Language)

Diseñado para almacenar y transportar información

Estándar en el intercambio de información estructurada entre diferentes plataformas.

Tiene:

* Extensibilidad
* Estructura
* Validación

Basado en texto

Orientado a los contenidos y no a la presentación

Las etiquetas se definen para crear los documentos, que no tiene significado establecido

No es sustituto de HTML

No existe un visor genérico de XML

SINTAXIS:

* Es case sensitive
* Todo elemento debe tener inicio y cierre o una etiqueda sola vacia.
* Todo elemento debe tener un elemento (raíz) que contenga a los demás.
* Todos los elementos deben estar correctamente anidados
* Todos los valores de los atributos deberían ir entre comilla

ESTRUCTURA

* Prologo poner que es xml, la versión, el encoding, y si es standalone. También el doctype con su dtd.
  + El dtd describe elementos, estructuras, contenido?
  + El xsd define el tipo de documento. Como dtd, pero mas completo. Incluye tipos de datos complejos.
* Cuerpo

**SOAP**

De la versión 0.9 a la 1.1

Simple Object Access Protocol

Desde la versión 1.2

Service Oriented Arquitecture Protocolo

Es un protocolo estándar en la capa de aplicación.

Basado en XML

Independiente de la plataforma

Mensajes se transportan generalmente por HTTP, pero aveces SMTP

SOAP MESSAGE

* SOAP ENVELOPE: Identifica al mensaje como SOAP
  + SOAP HEADER: Describe como manejar el mensaje
    - INFO
  + SOAP BODY: Contiene la información relativa a la llamada y respuesta
    - INFO
  + FAULT: Información de errores

WSDL

Xml de dfinicion de servicios web, tiene

<xml version>

<definitions>

<types> - los tipos de datos usados

<messages> - definición del metodo

<portType> - Define operaciones realizadas y los mensajes a los que tiene referencia

<binding> - Formato del mensaje y detalles del protocolo por cada portType

UDDI

Directorio de web services

Servicios web requieren:

Formato estándar para presentar datos: XML

Formato común para enviar mensajes: SOAP

Formato para definir servicios WSDL

Web services, siempre dos sistemas que se comunican entre si, el usuario no hace eso.

Web services tiene el problema que existe mucho procsamientos de datos y por lo tanto existe delay en la respuesta entre peticiones.

Servidor de Plaicaciones

Es un programa que provee infraestructura para servicios web

Dan

Clusterion,askdjalsdkdjlaskd

Estrategias para el desarrollo de web services:

TOP-DOWN

* Se dispone de un archivo WSDL para la generaci;on del servicio web.
* Sirve para hacer servicios web dummy (de prueba ) que sirve para garantizar que los cliente funcionan bien con el servidor

BOTTON – UP

* Exponer funcionalidades de sistema como un servicio web.
* Sirve para comprobar a través de los clientes que funcionan los servicios web.

JEE – Conjunto de Interfaces para hacer ciertas cosas.

Proyecto empresarial tiene

EJB

WEB

EAR - Integrador

**Bases de datos no SQL**

Tablas tradicionales pierden gran rendimiento al momento de tener una extensa cantidad de datos.

Grandes empresas requieren rendimiento, idearon SQL

Not only SQL. Cualquier Sistema de gestion de datos que no sigue el modelo de las bases de datos relacionales.

1998 por Carlo Strozzi.

En 2009 Eric Evans volvió a introducir el termino.

Evans dice Big Data, y Strozzi pide NoREL.

Use grandes volúmenes de datos y sistemas de datos distribuidos.

NoSQL no remplaza Bases de Datos Relacionales.

Historia:

1989 -> Surge base de datos documental llamada Lotus Domino

1990 -> MNesia basada en Erlang

2000 -> Neo4J basada en teoría de grafos

2003 -> Memcached

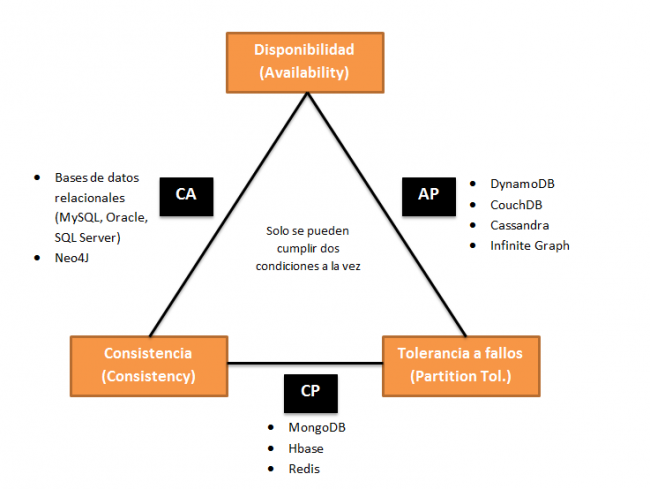
2008 -> Se abrieron las fuentes de Cassandra

2009 -> Finaliza el desarrollo inicial de MongoDB

TEOREMA CAP O BREWER

Es imposible para un sistema distribuida garantizar simultáneamente:

* Consistencia: Que todos los datos se muestren igual para todos.
* Disponibilidad: Que todos los recursos sean accesibles.
* Tolerancia del Particionado: Sigue funcionando así halla fallos.



No tiene garantía ACID: Atomicidad, Consistencia, Isolation (Aislamiento)(Integridad), Durabilidad. Hay una transacción cuando se garantiza ACID. Transacción o se cumplen todas o se niegan todos.

Commit -> Ejecutas transacción

Rollback -> Vuelve al estado anterior antes de la transacción

Esquema relacional este hecho a base de teoría de conjuntos.

No ofrecen un sentencia JOIN para combinar dos o mas registro de dos o mas tablas.

La estructura puede cambiar de forma dinámica sin afectar el resto. Generalmente no hacen validaciones a nivel de dato.

Características:

Escalabilidad Horizontal

Estructura Distribuida

Flexibilidad del Esquema

Consistencia Eventual – BASE

Tipos:

Documental -> Maneja structura de texto -> XML, JSON, BSON Ejemplos -> Elastic, MongoDB

Similares a las de clave valor, pero el valor es un documento. No tiene esquema estricto a seguir para definir. No permite JOINS.

Juegos en línea, comercio electrónico, almacenamiento de blogs, etc, Manejo de documentos y contenidos. NO tan bueno para transacciones

EJEMPLOS - mongoDB

Grafos

Inspiradas en la teoría de grafos de Euler. Modelo de datos de nodos: nodos, relaciones con pares clave/valor entre ambos.

Las relaciones pueden tener atributos y se puede hacer consulta a las relaciones.

Este tipo de base de datos solo se puede usar si la información se puede representar como una red.

Son eficientes para relacionar consultas donde existen relaciones de proximidad entre datos.

Se usa para sistema de recomendacionm BI, Análisis, Adminsitracion de sistemas, INdexacion de Datos, Logistica.

EJEMPLO – Neo4J

Clave/Valor – Clave Valor o Diccionario (Hashtable). Los valores no requieren de un esquema fijo. No existe el concepto de relaciones. Usada para consultar información básica.

Se puede usar cuando: Aplicaciónes pequeñas de lectura y escritura simples, datos escalares simples, carecen la capacidad de consulta avanzada.

Almacenamiento en cache de bases de datos relacionadas

Seguimientod e atributos transitorios.

Sessiones de usuario.

Almacenamiento.

EJEMPLOS – REDIS.

Familia de Columnas

Columnas

Radipa selecciona

Rápido cambios de equema

No para actualización constante

Usado para apps de inteligencia de negocio.

EJEMPLO - GOOGLE BIG TABLE, APACHE CASSANDRA,

Como escojer DB

A base de las necesidades de Cantidad de datos vs. Complejidad.

**SOA**

Arquitectura Orientada a Servicios.

A base de servicios web (SOAP y REST).

Arquitectura orientada a empresas grandes.

Se basa en que cada sistema, independientemente de como se lo realice, tienen su propio repositorio de datos, front y para compartir información exponen servicios. Los servicios expuestos solían ser SOAP, por medio de sockets, por medio de database link, archivos de texto, colas de mensajes.

Lo sistemas mas grandes que no necesariamente exponían servicios, podían consumir los servicios de otros sistemas para funcionar. La manera que consumían estos servicios, lo hacían por medio de un ESB (Enterprise Service Bus). Un ESB expone todos los servicios de otras aplicaciones y realiza la orquestación de servicios. Una orquestación de servicios es ejecutar servicios de diferentes lugares como si fueran uno solo logrando transaccionalidad.

**Grid Computing**

**Cluster**

Conjunto de varios servidores trabajando como uno solo.

Servidor de aplicaciones, Bases de datos mongo.

**Grid**

Conjunto de computadores que hacen una misma función, pero cada computadora es independiente, aun así ayudan en procesamiento.

**Cloud Computing**

Infraestructura en Internet. Modelos de negocio también

SaaS – Software en la nube (CRM) -> Salesforce

PaaS - Plataforma (Plataforma Sitio Web). Hosting. Heroku. Google Cloud Platform.

IaaS – Infraestructura (Esta en algun lugar del mundo). AWS

**NodeJS**