



# Análisis y Diseño de Sistemas 2

2015

Ing. Luis Alberto Arias Solórzano

Unidad 4

# REST

- ▶ **Representational State Transfer:** Es una técnica de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web. El término se originó en el año 2000, en una tesis doctoral sobre la web escrita por Roy Fielding, uno de los principales autores de la especificación del protocolo HTTP y ha pasado a ser ampliamente utilizado por la comunidad de desarrollo.
- ▶ El término REST en la actualidad se usa en un sentido más amplio para describir cualquier interfaz web simple que utiliza XML y HTTP, sin las abstracciones adicionales de los protocolos basados en patrones de intercambio de mensajes como el protocolo de servicios web SOAP.
- ▶ Es posible diseñar sistemas de servicios web de acuerdo con el estilo arquitectural REST de Fielding y también es posible diseñar interfaces XMLHTTP de acuerdo con el estilo de llamada a procedimiento remoto (RPC), pero sin usar SOAP.

# REST

- ▶ REST afirma que la web ha disfrutado de escalabilidad como resultado de una serie de diseños fundamentales clave:
- ▶ Un protocolo cliente/servidor sin estado: cada mensaje HTTP contiene toda la información necesaria para comprender la petición. Como resultado, ni el cliente ni el servidor necesitan recordar ningún estado de las comunicaciones entre mensajes. Sin embargo, en la práctica, muchas aplicaciones basadas en HTTP utilizan cookies y otros mecanismos para mantener el estado de la sesión.
- ▶ Un conjunto de operaciones bien definidas que se aplican a todos los recursos de información: HTTP en sí define un conjunto pequeño de operaciones, las más importantes son POST, GET, PUT y DELETE.
- ▶ Una sintaxis universal para identificar los recursos. En un sistema REST, cada recurso es direccionable únicamente a través de su URL.
- ▶ El uso de hipermedios, tanto para la información de la aplicación como para las transiciones de estado de la aplicación: la representación de este estado en un sistema REST son típicamente HTML o XML. Como resultado de esto, es posible navegar de un recurso REST a muchos otros, simplemente siguiendo enlaces sin requerir el uso de registros u otra infraestructura adicional.

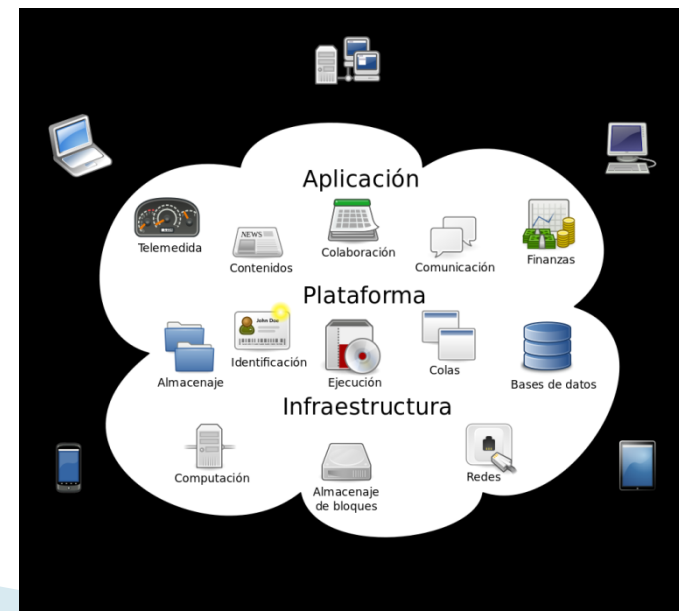
# REST (Ejemplos)

Dado que la definición de REST es muy amplia, es posible afirmar que existe un enorme número de aplicaciones REST en la red (prácticamente cualquier cosa accesible mediante una petición HTTP GET). De forma más restrictiva, en contraposición a los servicios web y el RPC, REST se puede encontrar en diferentes áreas de la web:

- ▶ La **blogosfera** –el universo de los blogs– está, en su mayor parte, basado en REST, dado que implica descargar ficheros XML (en formato RSS o Atom) que contienen listas de enlaces a otros recursos.
- ▶ **Amazon.com** ofrece su interfaz para desarrolladores tanto en formato REST como en formato SOAP (siendo la versión REST la que recibe mayor tráfico).
- ▶ **eBay** ofreció hasta 2008 una interfaz REST para desarrolladores.
- ▶ **Yahoo!** ofrece un API en REST para desarrolladores.
- ▶ El mecanismo de enrutamiento de Ruby on Rails soporta aplicaciones REST utilizando el patrón de diseño MVC.
- ▶ Microsoft tiene su implementación en **ADO.NET** Data Services Framework (anteriormente conocido como “Astoria”)
- ▶ El mismo mecanismo en **Catalyst** también soporta aplicaciones REST mediante MVC.
- ▶ Implementación REST para Java: **RestLet**.
- ▶ **Facebook** ofrece una API basado en REST.
- ▶ **Twitter** ofrece una API basado en REST.
- ▶ **MEGA** ofrece una API basado en REST.

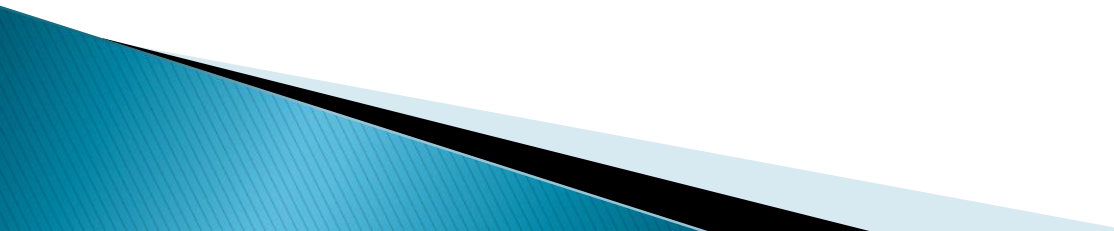
# Cloud Computing

- ▶ Es un nuevo modelo de prestación de servicios de negocio y tecnología, que permite incluso al usuario acceder a un catálogo de servicios estandarizados y responder con ellos a las necesidades de su negocio, de forma flexible y adaptativa, en caso de demandas no previsibles o de picos de trabajo, pagando únicamente por el consumo efectuado, o incluso gratuitamente en caso de proveedores que se financian mediante publicidad o de organizaciones sin ánimo de lucro.
- ▶ Según el IEEE Computer Society, es un paradigma en el que la información se almacena de manera permanente en servidores de Internet y se envía a cachés temporales de cliente, lo que incluye equipos de escritorio, centros de ocio, portátiles, etc.




# Cloud Computing

## Características:

- ▶ **Escalabilidad y elasticidad:** aprovisionamiento de recursos sobre una base de autoservicio en casi en tiempo real, sin que los usuarios necesiten cargas de alta duración.
  - ▶ **Virtualización:** permite compartir servidores y dispositivos de almacenamiento y una mayor utilización. Las aplicaciones pueden ser fácilmente migradas de un servidor físico a otro.
  - ▶ **Rendimiento:** Los sistemas en la nube controlan y optimizan el uso de los recursos de manera automática, dicha característica permite un seguimiento, control y notificación del mismo. Esta capacidad aporta transparencia tanto para el consumidor o el proveedor de servicio.
  - ▶ **Agilidad:** Capacidad de mejora para ofrecer recursos tecnológicos al usuario por parte del proveedor.
- 

# Cloud Computing

## Características:

- ▶ **Costo:** los proveedores en la nube afirman que los costos se reducen. Ello reduce barreras de entrada, ya que la infraestructura se proporciona típicamente por una tercera parte y no tiene que ser adquirida por un única sola vez o tareas informáticas intensivas infrecuentes.
  - ▶ **Acceso web:** permite a los usuarios acceder a los sistemas utilizando un navegador web, independientemente de su ubicación o del dispositivo que utilice (por ejemplo, PC, teléfono móvil).
  - ▶ **La seguridad:** puede mejorar debido a la centralización de los datos. La seguridad es a menudo tan buena o mejor que otros sistemas tradicionales, en parte porque los proveedores son capaces de dedicar recursos a la solución de los problemas de seguridad que muchos clientes no pueden permitirse.
  - ▶ **Mantenimiento:** de las aplicaciones de computación en la nube es más sencillo, ya que no necesitan ser instalados en el ordenador de cada usuario y se puede acceder desde diferentes lugares.
- 

# NoSQL

Es una amplia clase de sistemas de gestión de bases de datos que difieren del modelo clásico del sistema de gestión de bases de datos relacionales (RDBMS) en aspectos importantes, el más destacado es que no usan SQL como el principal lenguaje de consultas. Los datos almacenados no requieren estructuras fijas como tablas, normalmente no soportan operaciones JOIN, ni garantizan completamente ACID (atomicidad, consistencia, aislamiento y durabilidad), y habitualmente escalan bien horizontalmente.

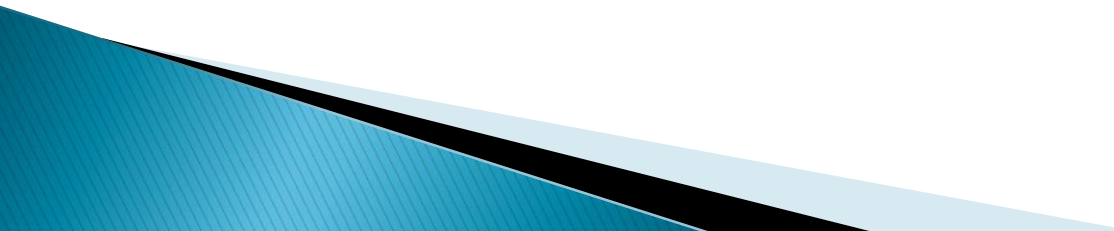
Se refieren a este tipo de bases de datos como almacenamiento estructurado, término que abarca también las bases de datos relacionales clásicas. A menudo, las bases de datos NoSQL se clasifican según su forma de almacenar los datos, y comprenden categorías como clave-valor, las implementaciones de BigTable, bases de datos documentales, y Bases de datos orientadas a grafos.



# NoSQL

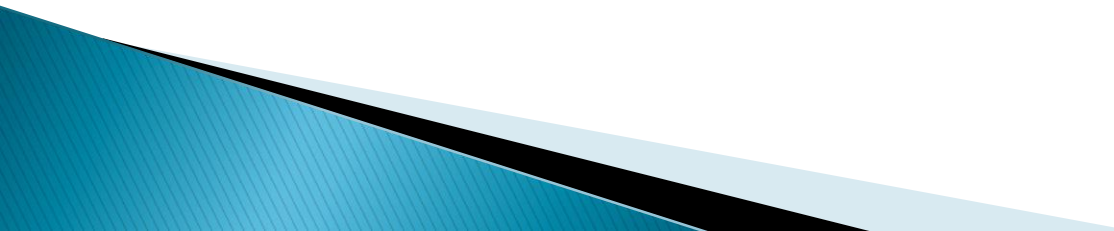
las bases de datos NoSQL están altamente optimizadas para las operaciones recuperar y agregar, y normalmente no ofrecen mucho más que la funcionalidad de almacenar los registros (p.ej. almacenamiento clave-valor). La pérdida de flexibilidad en tiempo de ejecución, comparado con los sistemas SQL clásicos, se ve compensada por ganancias significativas en escalabilidad y rendimiento cuando se trata con ciertos modelos de datos.

Los sistemas de bases de datos NoSQL crecieron con las principales compañías de Internet, como Google, Amazon, Twitter y Facebook. Estas tenían que enfrentarse a desafíos con el tratamiento de datos que las tradicionales RDBMS no solucionaban. Con el crecimiento de la web en tiempo real existía una necesidad de proporcionar información procesada a partir de grandes volúmenes de datos que tenían unas estructuras horizontales más o menos similares. Estas compañías se dieron cuenta de que el rendimiento y sus propiedades de tiempo real eran más importantes que la coherencia, en la que las bases de datos relacionales tradicionales dedicaban una gran cantidad de tiempo de proceso.



# NoSQL

## Ventajas

- ▶ Estos sistemas responden a las necesidades de escalabilidad horizontal que tienen cada vez más empresas.
  - ▶ Pueden manejar enormes cantidades de datos.
  - ▶ No generan cuellos de botella.
  - ▶ Escalamiento sencillo.
  - ▶ Diferentes DBs NoSQL para diferentes proyectos.
  - ▶ Se ejecutan en clusters de máquinas baratas.
- 

# NoSQL

## Bases de datos orientadas a objetos

- ▶ ObjectDB
- ▶ Zope Object Database
- ▶ db4o
- ▶ GemStone S
- ▶ Objectivity/DB

## Bases de datos documentales

- ▶ CouchDB, de Apache CouchDB
- ▶ MongoDB, de 10gen
- ▶ RavenDB, de Hibernate Rhinos.
- ▶ BaseX
- ▶ djondb
- ▶ eXist
- ▶ SimpleDB, de Amazon
- ▶ IBM Lotus Domino
- ▶ Terrastore

## Bases de datos en grafo

- ▶ Neo4j
- ▶ DEX/Sparksee
- ▶ AllegroGraph
- ▶ OrientDB
- ▶ InfiniteGraph
- ▶ Sones GraphDB
- ▶ InfoGrid
- ▶ HyperGraphDB

# NoSQL

## Bases de datos clave/valor

- ▶ Cassandra, de Apache The Apache Cassandra
- ▶ BigTable, de Google
- ▶ Dynamo, de Amazon
- ▶ MongoDB
- ▶ Project Voldemort, de LinkedIn
- ▶ Riak
- ▶ Redis
- ▶ Oracle NoSQL

## Bases de datos multivalor

- ▶ OpenQM
- ▶ Extensible storage engine

## Bases de datos tabular

- ▶ HBase, de Apache
- ▶ BigTable, de Google
- ▶ LevelDB, versión abierta de BigTable
- ▶ Hypertable

**Gracias**