Índice

- Comunicaciones en sistemas distribuidos (SOA)
  - SOAP
  - REST
- Web Semántica
- SPARQL

- Certificaciones
- Caso Práctico: Proyecto en Heroku

SOA (Service Oriented Architecture)

- La Arquitectura Orientada a Servicios es un **concepto de arquitectura de software que define la utilización de servicios para dar soporte a los requisitos del negocio**.
  - Permite la creación de sistemas altamente escalables que reflejan el negocio de la organización, a su vez brinda **una forma estándar de exposición e invocación de servicios** (comúnmente pero no exclusivamente servicios web), lo cual facilita la interacción entre diferentes sistemas propios o de terceros.
  - SOA define las siguientes capas de software:
    - Aplicaciones básicas - Sistemas desarrollados bajo cualquier arquitectura o tecnología, geográficamente dispersos y bajo cualquier figura de propiedad;
    - De exposición de funcionalidades - Donde las funcionalidades de la capa aplicativas son expuestas en forma de servicios (servicios web);
    - De integración de servicios - Facilitan el intercambio de datos entre elementos de la capa aplicativa orientada a procesos empresariales internos o en colaboración;
    - De composición de procesos - Que define el proceso en términos del negocio y sus necesidades, y que varia en función del negocio;
    - De entrega - donde los servicios son desplegados a los usuarios finales.
  - SOA proporciona una **metodología y un marco de trabajo** para documentar las capacidades de negocio y puede dar soporte a las actividades de integración y consolidación.

Definiciones SOA

Término	Definción/Comentario
Servicio	Una **función** sin estado (Existen servicios asíncronos en los que una solicitud a un servicio crea, por ejemplo, un archivo, y en una segunda solicitud se obtiene ese archivo), auto-contenida, **que acepta una(s) llamada(s) y devuelve una(s) respuesta(s) mediante una interfaz bien definida**. Los servicios pueden también ejecutar unidades discretas de trabajo como serían editar y procesar una transacción. Los servicios no dependen del estado de otras funciones o procesos. La tecnología concreta utilizada para prestar el servicio no es parte de esta definición.
Orquestación	**Secuenciar los servicios y proveer la lógica adicional** para procesar datos. No incluye la presentación de los datos. Coordinación.
Sin estado	**No mantiene ni depende de condición pre-existente alguna.** En una SOA los servicios no son dependientes de la condición de ningún otro servicio. Reciben en la llamada toda la información que necesitan para dar una respuesta. Debido a que los servicios son "sin estado", pueden ser secuenciados (orquestados) en numerosas secuencias (algunas veces llamadas tuberías o pipelines) para realizar la lógica del negocio.
Proveedor	La función que brinda un servicio en respuesta a una llamada o petición desde un consumidor.
Consumidor	La función que consume el resultado del servicio provisto por un proveedor.

SOA frente OOA

- Al contrario de las arquitecturas orientadas a objetos, las SOAs están formadas por **servicios de aplicación débilmente acoplados y altamente interoperables**.
- Para comunicarse entre sí, estos servicios se basan en una definición formal independiente de la plataforma subyacente y del lenguaje de programación (p.ej., WSDL). **La definición de la interfaz encapsula (oculta) las particularidades de una implementación, lo que la hace independiente del fabricante, del lenguaje de programación o de la tecnología de desarrollo (como Plataforma Java o Microsoft.NET).**
- Con esta arquitectura, se pretende que los **componentes software desarrollados sean muy reusables**, ya que la interfaz se define siguiendo un estándar; así, un servicio C Sharp podría ser usado por una aplicación Java.

Composición de servicios

- Composición de servicios se refiere a la **creación de nuevos servicios o instancias de servicios mediante otros servicios o componentes de servicios existentes**.
- El nivel de composición de servicios incluye la composición de sesiones de servicio, recursos de servicio o componentes
- La composición de servicios en tiempo de ejecución incluye:
  - **Composición estática** o predefinida entre componentes del nivel de servicio
  - **Composición dinámica** entre componentes del nivel de servicio

Web Services

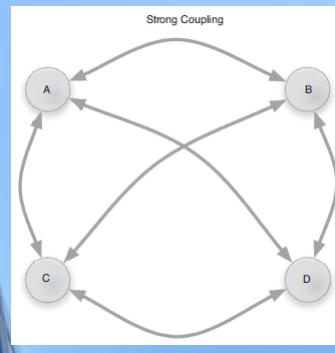
- Los Web Services son un tipo de servicios que usan sus propias interfaces programa-programa, protocolos y servicios de registro de tal manera que **posibilitan que aplicaciones de diferentes plataformas puedan comunicarse con otras aplicaciones**.
- Un Web Service se aprovecha de la **especificación de XML** para definir tanto su descripción, como los mensajes que recibe y produce al igual que en los servicios de registro del servicio.

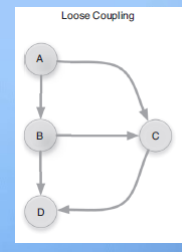
Web Services

- La idea general alrededor de Web Services no es algo nuevo.
- Antecesores en el tema de computación distribuida han existido y funcionado:
  RPC, EDI, CORBA, COM, APPC
- La diferencia se encuentra en el **alto nivel de encapsulación e independencia entre las aplicaciones**.
- El gestor de la iniciativa es W3C, garantía de evolución y futuro.

Alto acoplamiento vs desacoplo

- Acoplamiento: Grado de conocimiento directo de un componente hacia otro.
- Afecta a la fiabilidad. Necesario para las aplicaciones Web
- También al mantenimiento: Un cambio en un módulo implica cambios en todos los módulos dependientes

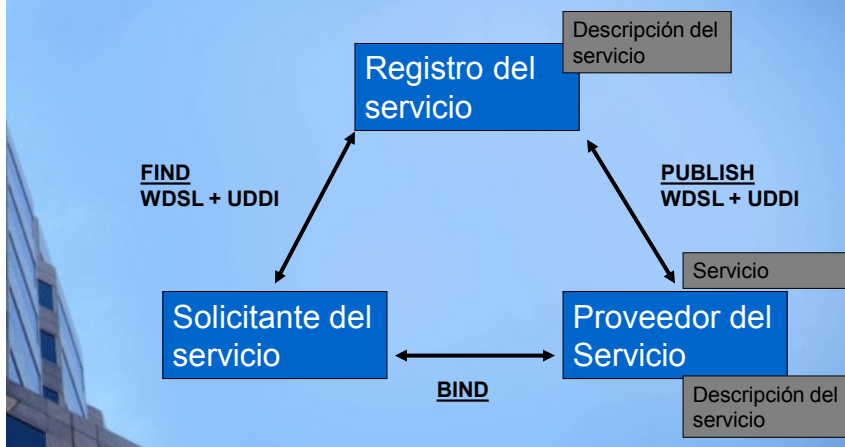




Web Services: Propiedades

- Deben tener una interfaz pública definida en XML. Esta interfaz describe todos los **métodos** disponibles a los clientes y especifica la **signatura** para cada método
- La definición de esta interfaz se hace con el **lenguaje WSDL (Web Service Description Language)**
- Existe una forma de **localizar el servicio** y su interfaz WSDL, **a través de UDDI** (Universal Description, Discovery, and Integration)

Web Services: Propiedades

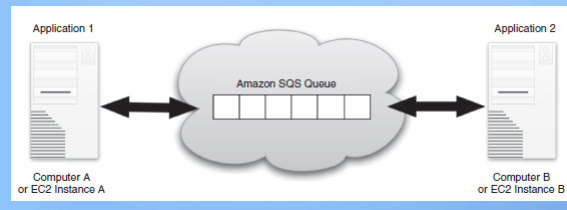


SOA y Cloud Computing

- Son complementarios: SOA define cómo debe ser una arquitectura y Cloud computing cómo debe desplegarse.
- Se combinan juntos: Despliegues de servicios SOA + Escalabilidad y economía Cloud
- ¿Cómo se produce la comunicación con los servicios en Cloud?
  - Directamente invocando el servicio Web de la aplicación con la que nos queremos comunicar.
  - Utilizando un Middleware de comunicaciones.

SOA y Cloud Computing

- Se puede utilizar un middleware de mensajes como el Amazon Simple Queue Service.
- Este servicio utiliza WebServices.
- Permite comunicaciones asíncronas y almacenamiento persistente en la cola de mensajes.



APIs web: SOAP vs. REST

SOAP: Simple Object Access Protocol

- **Define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML**.
  - Deriva de un protocolo creado por David Winer en 1998, llamado XML-RPC.
- Fue creado por Microsoft, IBM, y otros y actualmente se encuentra bajo el auspicio de la W3C.
- Su arquitectura consiste en varias capas de especificaciones para formato de mensajes:
  - Message Exchange Patterns (MEP)
  - Protocolos de transporte (SMTP y HTTP/S)
  - Modelos de procesado de mensajes
  - Protocolo de extensibilidad
  - WS-*

REST: Representational State Transfer

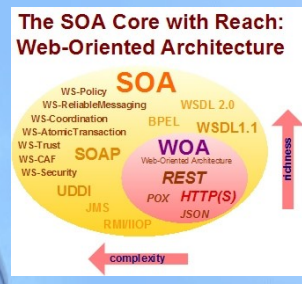
- **Estilo de arquitectura** software para sistemas Web dsitribuidos.
  - Introducido en la tésis doctoral de **Roy Fielding** en el año 2000.
- Se refiere a una colección de principios de arquitectura de red, que **marcan cómo definir e invocar los recursos**.
- El término se usa a veces para describir una simple interfaz que transmite datos de un dominio específico por **HTTP** sin capas adicionales como SOAP o uso de cookies.
- Los sistemas que cumplen los principios marcados por Fielding suelen ser referidos como sistemas **RESTful**.

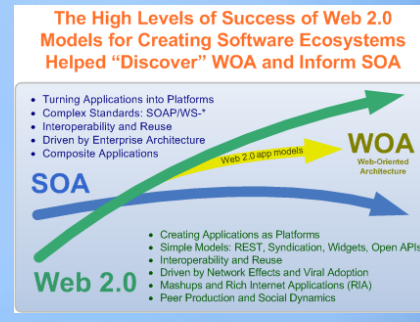
SOA vs WOA

- WOA: Web Oriented Architecture: SOA + REST + WWW

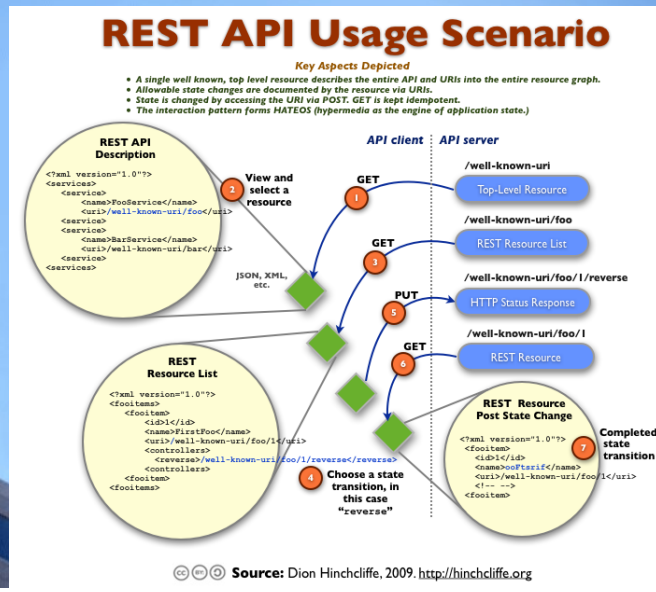
- Arquitectura de software que extiende SOA a aplicaciones basadas en web
- **La información se representa en forma de recursos en la red** y es accedida y manipulada mediante el protocolo especificado en la URI.
- Los recursos se manipulan mediante CRUD **HTTP (POST, GET, PUT, DELETE) usando REST.**
- Estos recursos sólo son manipulados por componentes pertenecientes a la red (esencialmente browsers & web servers).
- Es responsabilidad de los componentes entender la representación y estados de transición válidos de los recursos.
- Los recursos tienen embebidas URIs que construyen una red más grande de recursos
- Más flexibles, ágiles al cambio y evolución de los sistemas de información.

SOA vs WOA





Usando REST

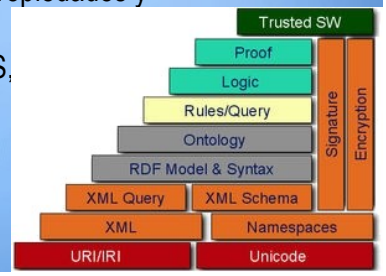


Web Semántica

- Problema de la Web Actual:
  - El significado de la web no es comprensible por máquinas
- **Web Semántica** → crea un medio universal de intercambio de información, aportando semántica a los documentos en la web
  - Añade significado comprensible por ordenadores a la Web
  - Usa técnicas inteligentes que explotan esa semántica
  - Liderada por Tim Berners-Lee del W3C
- **Misión** → "turning existing web content into machine-readable content"

Web Semántica

- La Web Semántica está compuesta de:
  - **XML, sintaxis para documentos estructurados**
  - XML Schema, restringe la estructura de documentos XML
  - **RDF es un modelo de datos que hace referencia a objetos y sus relaciones**
  - RDF Schema, vocabulario para definir propiedades y clases de recursos RDF
  - **OWL**, añade más vocabulario que RDFS, entre clases, cardinalidad, igualdad …



Web Semántica: RDF

- RDF identifica conceptos usando identificadores Web (URIs), y describe recursos con propiedades y valores de las mismas
- Definiciones:
  - Un **Recurso** es cualquier cosa que puede tener una URI, como por ejemplo "http://www.w3schools.com/RDF"
  - Una **Propiedad** es un Recurso que tiene un nombre, como “autor" o “páginaweb"
  - Un **Valor de propiedad** es el valor de una Propiedad, tal como “Autor1" o "http://www.w3schools.com" (un valor de propiedad puede corresponder a un recurso)

Web Semántica: RDF

- Un **grafo RDF crea una web de conceptos distribuidos**
  - Realiza aserciones sobre relaciones lógicas entre entidades
  - La información en RDF puede ligarse con grafos en otros lugares
  - Mediante software se pueden realizar inferencias
  - Existen **lenguajes de consulta** sobre triple stores como SPARQL
- Mediante RDF hacemos que la información sea procesable por máquinas
  - Agentes software pueden guardar, intercambiar y utilizar metadatos sobre recursos en la web
- **Ontología** → jerarquía de términos a utilizar en etiquetado de recursos → formalización de los metadatos de un dominio.

Web Semántica: RDF

- **Formato RDF/XML:**

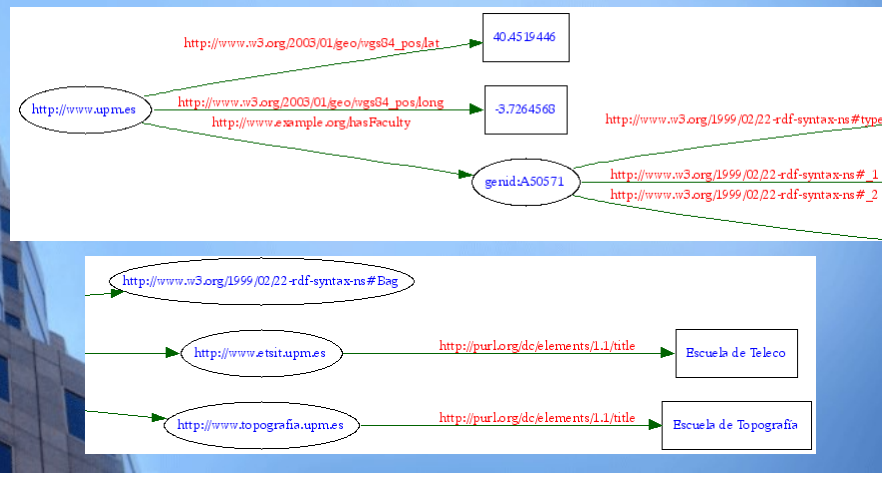
```
1: <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
2:    xmlns:dc="http://purl.org/dc/elements/1.1/"
3:    xmlns:geo="http://www.w3.org/2003/01/geo/wgs84_pos/"
4:    xmlns:edu="http://www.example.org/">
5:    <rdf:Description rdf:about="http://www.upm.es">
6:       <geo:lat>40.4519446</geo:lat>
7:       <geo:long>-3.7264568</geo:long>
8:       <edu:hasFaculty>
9:          <rdf:Bag>
10:            <rdf:li rdf:resource="http://www.etsit.upm.es" dc:title="Escuela de Teleco"/>
11:            <rdf:li rdf:resource="http://www.topografia.upm.es" dc:title="Escuela de Topografía"/>
12:            </rdf:Bag>
13:      </edu:hasFaculty>
14:   </rdf:Description>
15: </rdf:RDF>
```

- **Formato: N3/Turtle:**

```
1: @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2: @prefix dc: <http://purl.org/dc/elements/1.1/> .
3: @prefix geo: <http://www. w3.org/2003/01/geo/wgs84_pos#> .
4: @prefix edu: <http://www.example.org/> .
5: <http://www.upm.es> geo:lat "40.4519446" ; geo:long "-3.7264568" .
6: <http://www.upm.es> edu:hasFaculty <http://www.etsit.upm.es> .
7: <http://www.etsit.upm.es> dc:title "Escuela de Teleco" .
8: <http://www.upm.es> edu:hasFaculty <http://www.topografia.upm.es> .
9: <http://www.etsit.upm.es> dc:title "Escuela de Topografía" .
```

Web Semántica: RDF

- **Validamos con: http://www.w3.org/RDF/Validator/**



Web Semántica: Ontologías

- Una ontología define conceptos de un dominio y relaciones entre ellos
- Los bloques básicos que componen el diseño de una ontología son:
  - clases o conceptos
  - propiedades de cada concepto describiendo varias características y atributos del concepto
  - restricciones sobre las propiedades
- Una **ontología junto con** las **instancias de sus clases** individuales constituyen un **knowledge base**

Web Semántica: Ontologías

- Una ontología difiere de un esquema XML en que es **una representación de conocimiento**, no un formato de mensaje
- La principal ventaja de una ontología escrita en OWL es que **hay disponibles herramientas que pueden razonar sobre ella**
- La sintaxis de intercambio de información en OWL es normalmente RDF/XML.
- OWL es una extensión del vocabulario de RDF
- Las ontologías Web son distribuidas
- Pueden ser importadas y extendidas para crear ontologías derivadas
- Se pueden alinear unas ontologías con otras

Web Semántica: Razonamiento

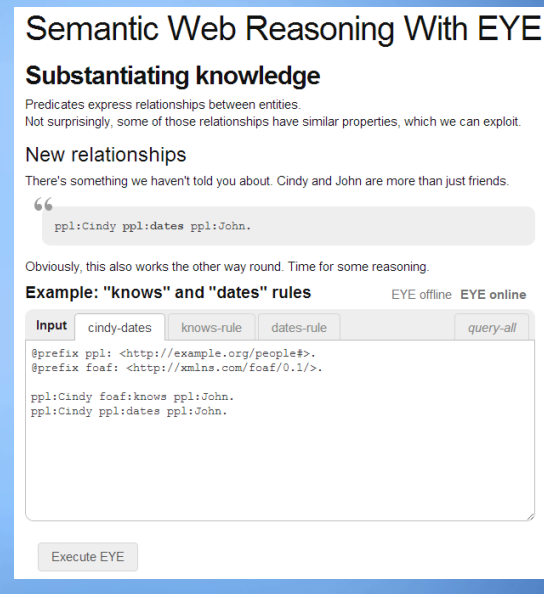
```
@prefix ppl: <http://example.org/people#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>.
@prefix owl: <http://www.w3.org/2002/07/owl#>.
ppl:Cindy foaf:knows ppl:John.
ppl:Cindy ppl:dates ppl:John.

foaf:knows a owl:SymmetricProperty.
ppl:dates a owl:SymmetricProperty.


{?P a owl:SymmetricProperty. ?S ?P ?O} => {?O ?P ?S}.
```

RESULTADO:

```
ppl:Cindy foaf:knows ppl:John.
ppl:Cindy ppl:dates ppl:John.
foaf:knows a owl:SymmetricProperty.
ppl:dates a owl:SymmetricProperty.
ppl:John foaf:knows ppl:Cindy.
ppl:John ppl:dates ppl:Cindy.
```

Web Semántica: Razonamiento

Razonamiento semántico en Web con EYE: http://n3.restdesc.org/



SPARQL

- SPARQL (http://www.w3.org/TR/rdf-sparql-query/) permite la consulta de grafos RDF a través de un lenguaje sencillo
- SPARQL es idóneo para extraer y consultar información mantenida por aplicaciones, servicios o repositorios ad-hoc de terceras partes expresados en RDF

"Find me the capital of all countries in Africa"

```
PREFIX abc: <nul://sparql/exampleOntology#> .
SELECT ?capital ?country
WHERE {
  ?x abc:cityname ?capital ;
     abc:isCapitalOf ?y.
  ?y abc:countryname ?country ;
     abc:isInContinent abc:Africa.
}
```

GeoSPARQL

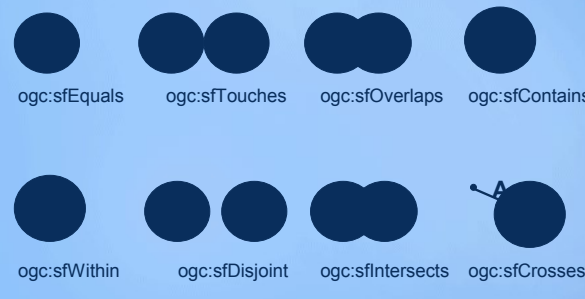
- Extensión de SPARQL para soportar datos geoespaciales.
- Proporciona una pequeña ontología en RDFS/OWL para utilizar GML

Example Query: Find airports within 100 KM of Reston, VA

```
SELECT ?airport
WHERE { ?airport rdf:type :Airport .
        ?airport :hasPointGeometry [
                 ogc:asWKT ?aPointGeom ]
        FILTER(ogcf:distance(?aPointGeom,
                 "POINT(-77.2 38.57)"^^ogc:WKTLiteral,
                  ogc:km) <= 100) }
```

GeoSPARQL

- Relaciones Topológicas



LinkedData

- "A term used to describe a recommended best practice for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs and RDF."
- Permite descubrir, conectar, describir y reutilizar todo tipo de datos.
  - Pasa de una Web de Documentos a una Web de Datos
    - En Mayo 2009 ya contiene 4,2 billones de tripletas RDF, ligadas por 142 millones de enlaces
- Pensado para abrir y conectar diversos vocabularios e instancias semánticas, para que puedan ser utilizados por la comunidad semántica
- URL: http://linkeddata.org/

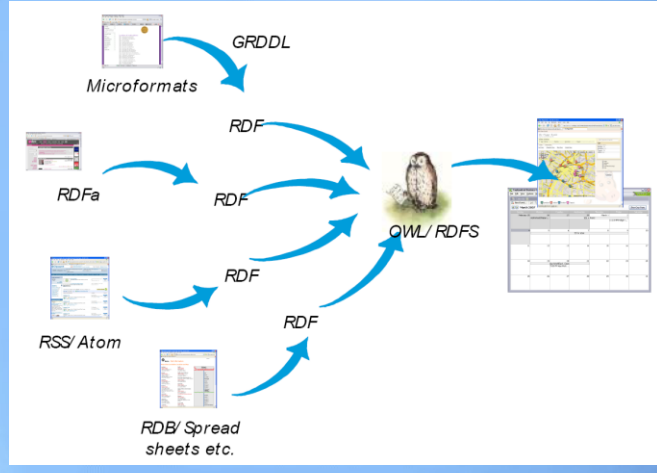
Mashups y mashups semánticos

- En los mash-ups Web 2.0 tradicionales, **cada vez que se integra una nueva fuente de información es preciso desarrollar un nuevo adaptador** que convierta los datos capturados al formato interno utilizado en el mash-up.
- Para evitar esto se utiliza RDF y OWL, que permiten combinar las respuestas de un portal con las de otro
  - haciendo que las sentencias RDF de dos localizaciones diferentes hagan referencia al mismo concepto (URI)
  - estableciendo correspondencias mediante OWL indicando que dos conceptos son equivalentes

Mashups y mashups semánticos

- Los **mash-up semánticos son mucho más flexibles** – convierten la información recuperada a formato RDF (*lingua franca*) fácilmente filtrable y consultable con SPARQL.
  - **Tienen la capacidad de evolucionar** sin requerir cambios en su código
    - Los datos provistos en formatos de representación sintácticos diferentes, pero tales **datos, semánticamente, deben proveer una información muy similar**, fácilmente convertible a un vocabulario RDF común.
- Ejemplo:
  - Mash-up semántico capaz de agregar información heterogénea sobre eventos provenientes de diferentes organizaciones y de mostrarla de manera conjunta sobre un mapa de GoogleMaps

Mashups y mashups semánticos



Conclusiones

- La **Web del Futuro será una plataforma de ejecución de servicios cada vez más inteligentes, consumibles y alojados en dispositivos heterogéneos** (desde la nube, a servidores web tradicionales o los propios objetos cotidianos)
- Los paradigmas de *Web Oriented Architecture* y *Semantic Web* están influyendo en las soluciones Cloud y su despliegue.
  - Poco a poco llegaremos a una situación en la que **todo se aloje en la Web**. El acceso, modificación de información y la gestión de los recursos se realizarán a través del navegador.
  - La coordinación de Web de Datos y los Ecosistemas de Servicios Distribuidos en Internet se realizará mediante **mediación semántica.**

Caso práctico: Proyecto en heroku

- Propiedad de Salesforce
- Servicio de Hosting en la nube (PaaS)
- Gratuito (hasta 5 MB de espacio en disco para base de datos, 200MB para todos los archivos incluyendo repositorios Git)
- Incluye base de datos postgres 10K filas
- Gestión de código a través de Git

Caso práctico: Proyecto en heroku

- Procesamiento basado en Dynos
- Dynos: Son servidores ligeros
- A single Heroku dyno provides 512mb ram.
- It has 4 (virtual) CPU cores (Intel Xeon X5550 @ 2.67GHz).
- Precio de un dyno: 0.05 $/hora

Caso práctico: Proyecto en heroku

- Almacenamiento en Bases de Datos:
- Heroku puede trabajar con MySQL, SQLite, PostgreSQL, MongoDB, CouchDB y Memcache a traves de un tercero.
- Mongo HQ y Cloudant (como add-ons)

Caso práctico: Proyecto en heroku

- Aplicaciones que corren en Heroku

- Best Buy (IdeaX)
- Shopify.com
- Scrumninja.com
- Flightcaster.com

Caso práctico: Proyecto en heroku

http://smarket.herokuapp.com/

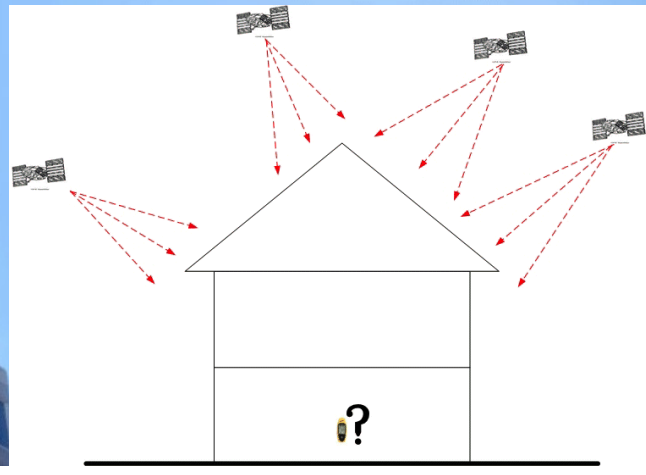
SMARKET

Caso práctico: Proyecto en heroku

- SMARKET- Seguimiento Espacial - Propósito
  - Servicios de ruta óptima en el proceso de compra.
  - Información sobre **ofertas personalizadas** al perfil de usuario en cuestión. Podrán generarse **alertas cuando el cliente se encuentre en un pasillo o zona en la que tiene descuentos**.
  - **Servicios de navegación**.
  - Información sobre el estado de las cajas para determinar en la que el tiempo de espera será menor.

Caso práctico: Proyecto en heroku

- SMARKET- Seguimiento Espacial
  - Problemática



Caso práctico: Proyecto en heroku

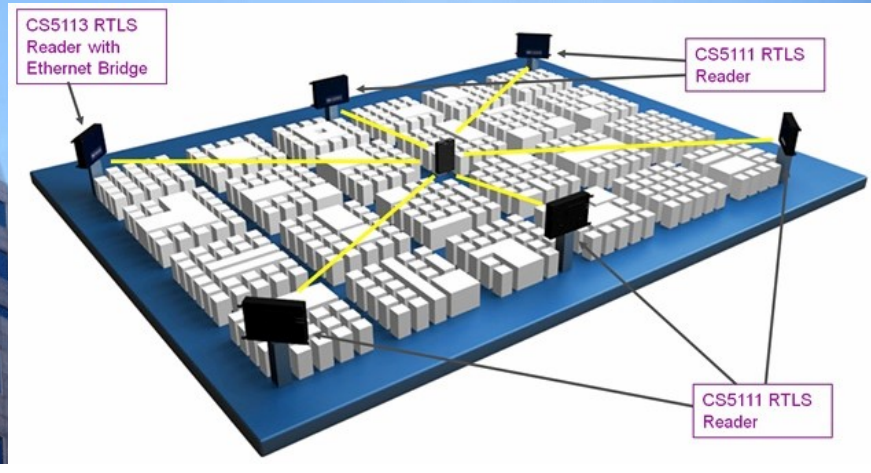
- SMARKET- Seguimiento Espacial
  - Problemática

Tecnologías de Localización de Interiores

Wi-Fi positioning	Cellular Methods	Sensor Tracking	Marcadores de Radiofrecuencia (RFID)	NFC	Ópticos

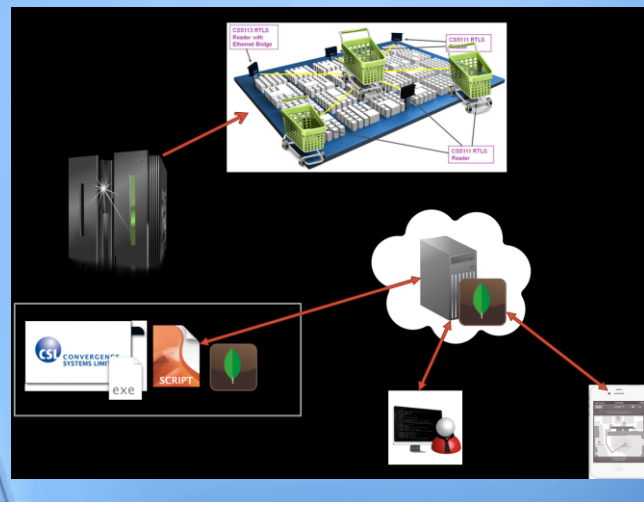
Caso práctico: Proyecto en heroku

- SMARKET- Seguimiento Espacial
  - Problemática



Caso práctico: Proyecto en heroku

- SMARKET- Seguimiento Espacial



Caso práctico: Proyecto en heroku

•Node.js

Entorno de programación multiplataforma diseñado para escribir aplicaciones de Internet altamente escalables, especialmente servidores Web.

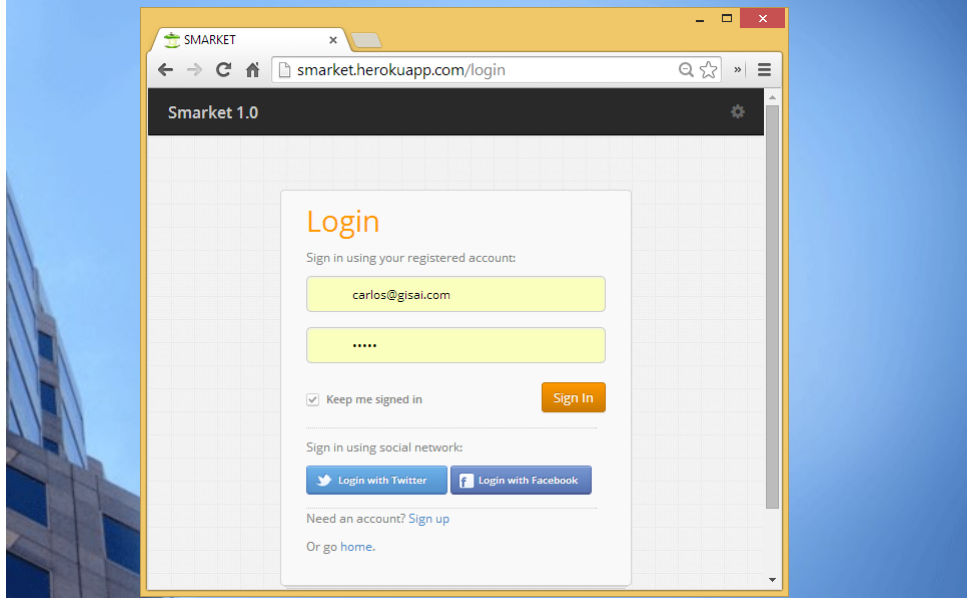
Características	Casos de uso típicos
•JavaScript •Orientado a eventos •Entrada/Salida asíncrona	•API RESTful •Herramientas de colaboración •Tiempo real •Bases de datos

•MongoDB

Sistema de base datos no relacional (NoSQL) orientado a documentos.

Características
•JSON •Consultas *adhoc* •Indexación •Replicación •JavaScript

Caso práctico: Proyecto en heroku



Caso práctico: Proyecto en heroku