Introducción a WS-REST

Ing. Guillermo Roldós

Agosto 2010

Agenda

- Descripción general
- Arquitectura orientada a recursos (ROA)
- Soporte Java y .NET
- Calidad de servicio
- Casos de estudio
  - Dominios de aplicación
  - Integración con otras tecnologías

Descripción General

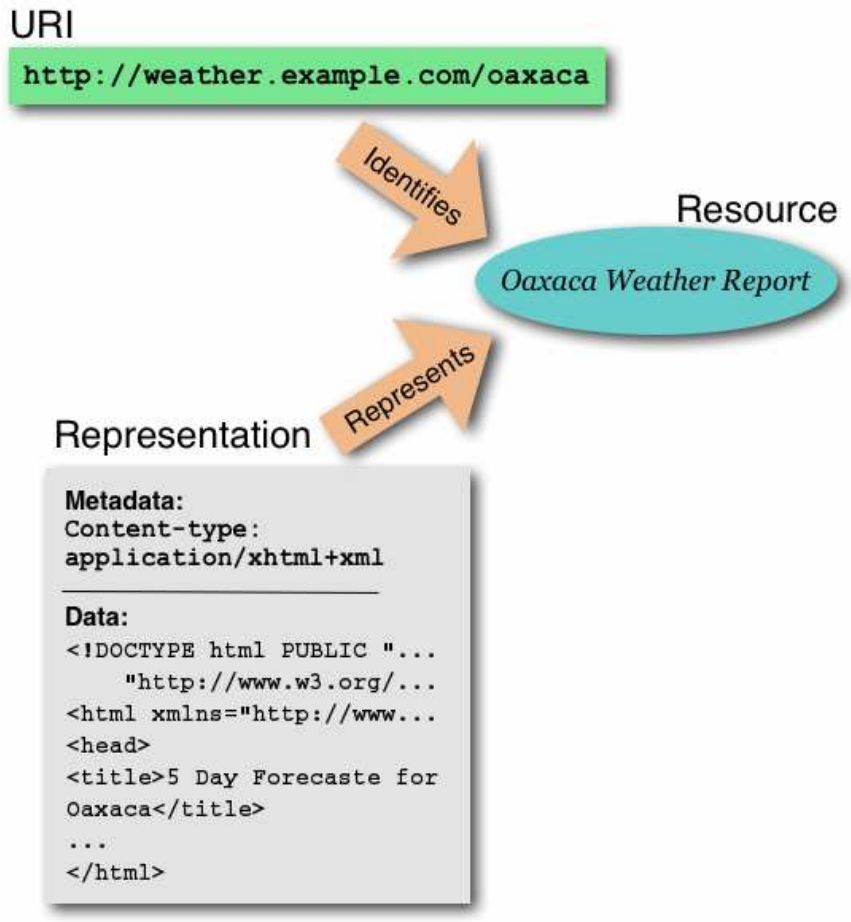
- Por qué surge REST
  - WS-* es muy completo pero muy complejo
- Es un estilo de arquitectura, no es una especificación ni un estándar
- Surge de la tesis de doctorado de Roy Fielding
  - Define restricciones para que un sistema sea REST
  - Se basa en la Web estática

Descripción General

- Principios o Restricciones
  - Recursos deben ser uniformemente accesibles (URI única)
  - Recursos son accedidos y actualizados por operaciones GET, POST, PUT, DELETE
  - Metadatos para describir recursos
  - Comunicación entre cliente y servidor debe ser *stateless*

Descripción General

- Recursos
  - Cualquier cosa que pueda identificar usando una URI
- URI
  - Dirección universal a un recurso
- Representación
  - Lo que se envía al cliente cuando solicita un recurso



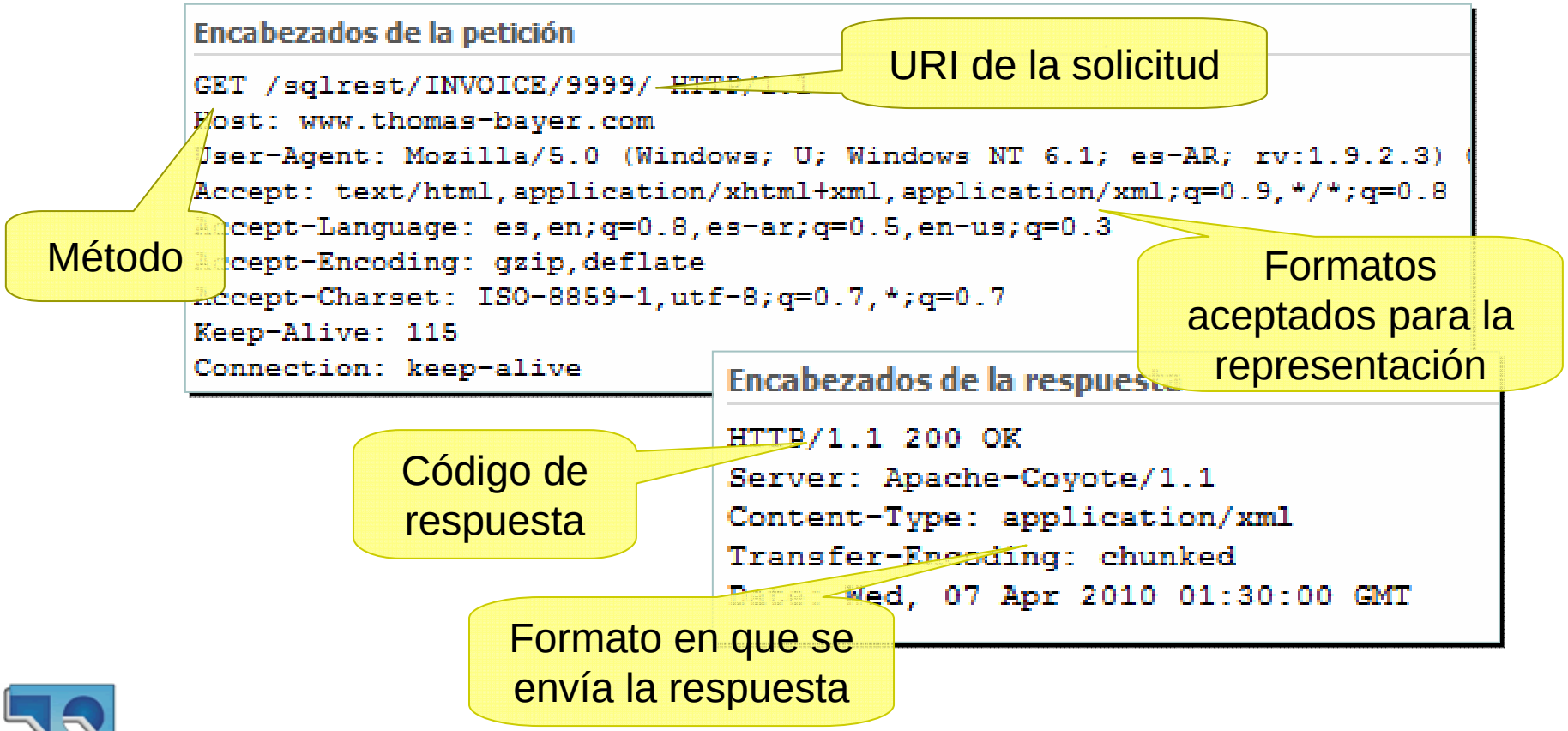
Descripción General

- Funcionamiento
  - Orientado a recursos, no a operaciones
  - Una invocación REST es un *request* HTTP
  - Las operaciones sobre recursos son limitadas

Operación	Método HTTP
Create	POST
Retrieve	GET
Update	PUT
Delete	DELETE

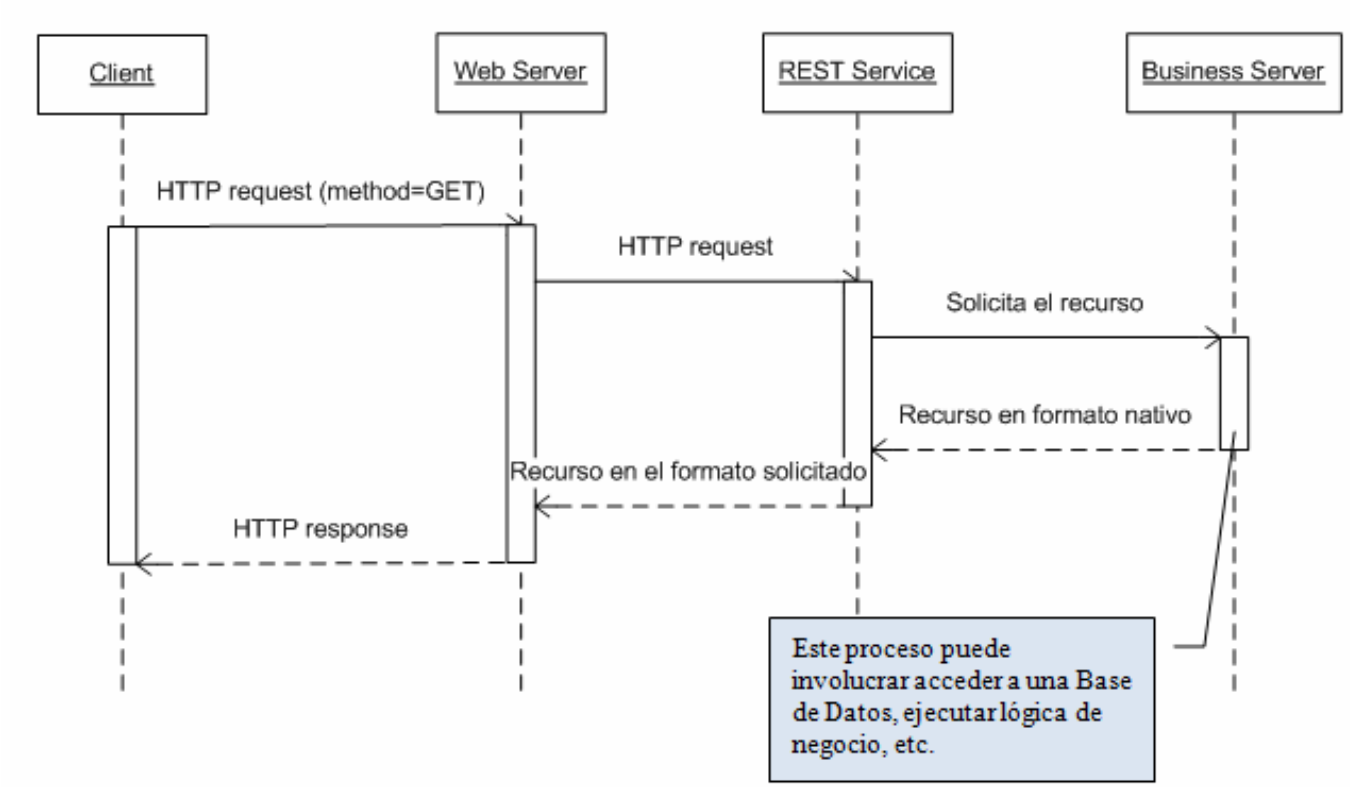
Descripción General

- Ejemplo de invocación REST



Descripción General

- Que ocurrió en el servidor



Descripción General

- Servicios REST híbridos
  - Servicios que dicen ser REST pero no lo son totalmente
  - Ejemplo: servicio Flickr de Yahoo

Llamada RPC con apariencia REST

```
http://api.flickr.com/services/rest?method=flickr.g
  alleries.getList&api_key=1111

http://api.flickr.com/services/rest?method=flickr.g
  alleries.create&api_key=1111
```

Se crean recursos usando método GET

Descripción General

- REST vs WS-*

REST	WS-*
Orientado a recursos	Orientado a operaciones
Busca escalabilidad y performance	Busca interoperabilidad
Ve la Web como un gran repositorio	Ve la Web como medio para intercambio de mensajes
Conjunto limitado de operaciones	Operaciones ilimitadas
Servidor *stateless*	*Stateless* o *stateful*
Navegación a través de links	Grafo de estados debe conocerse
No hay estándares de seguridad, transaccionalidad, etc.	Existe stack WS-*

Descripción General

- Conclusiones
  - Ninguno es mejor que el otro a priori
  - SOAP es mejor para integrar Sistemas heterogéneos, código legado, etc.
  - REST es mejor para aplicaciones Web dirigidas a clientes desconocidos

Arquitectura orientada a recursos

- Es importante formalizar una arquitectura
- Basado en axiomas
  - Los recursos son abstractos y se pueden representar por al menos una URI
  - Las representaciones son inmutables
  - Existe al menos una URI para cada recurso
  - Cada solicitud al servidor es independiente de todas las otras

Arquitectura orientada a recursos

- Navegabilidad
  - Brindada a través de links en las propias representaciones
- Interface única
  - Usar métodos GET, POST, PUT, DELETE
  - No usar GET para crear recursos
  - En lo posible usar operaciones idempotentes

Arquitectura orientada a recursos

- Guía para el diseño de un servicio REST
  - Identificar los recursos del Sistema
  - Diseñar las URIs
  - Definir las operaciones disponibles sobre cada recurso
  - Definir tipos de representación que se aceptarán
  - Definir un esquema de seguridad

Soporte Java

- Especificación JAX-RS
  - Usa *annotations* sobre clases POJO
  - Es posible especificar recursos, *paths*, operaciones, etc.

```
@Path("/printers")
public class PrintersResource {
    @GET
    public WebResourceList getMyResources() { ... }

    @GET @Path("/list")
    public WebResourceList getListOfPrinters() { ... }
}
```

Arquitectura orientada a recursos

- Implementaciones de JAX-RS
  - Jersey es la implementación de referencia
  - RESTEasy es provista por JBoss
    - Se integra con otros *containers*
    - Permite la integración con EJB y Spring
    - Incluye un *Client Framework* para simplificar el desarrollo de clientes

Soporte .NET

- Incluido en *Windows Communication Foundation*
  - A través del *WCF REST Programming Model*
  - Se mapean los servicios expuestos y su implementación

```
[ServiceContract]
interface ICustomer
{
  [OperationContract]
  [WebGet( UriTemplate="customers/{id}" )]
  Customer GetCustomer( string id ):

  [OperationContract]
  [WebInvoke( UriTemplate="customers/{id}", Method="PUT" )]
  Customer UpdateCustomer( string id, Customer newCustomer );
}
```

Soporte .NET

- WCF REST Starter Kit
  - A partir de la versión 3.5 SP1 del .NET Framework
  - Incluye nuevos *templates* para Visual Studio
  - Otras funcionalidades
    - Página de ayuda autogenerada
    - Manejo de excepciones
    - Soporte de cache
    - Desarrollo de clientes HTTP

Calidad de servicio

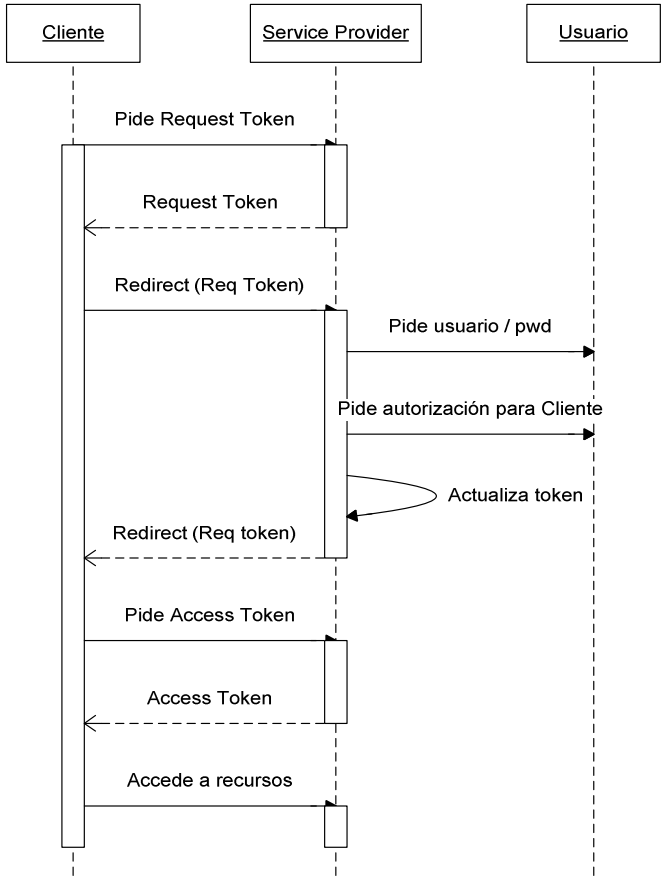
- Especificaciones REST-*
  - Buscan emular a WS-*
  - Muy inmaduras e incompletas
  - Mucho campo para aportar

Calidad de servicio

- Especificaciones REST-* con avances
  - Mensajería
  - Transacciones
- Especificaciones REST-* sin avances
  - Workflow y BPM
  - Caching
  - Compensaciones

Calidad de servicio

- O-Auth
  - Estándar que permite compartir recursos
  - El dueño no tiene que compartir credenciales



Dominios de aplicación

- e-Commerce
  - Todos los sitios proveen interfaces REST
  - La gran mayoría son híbridas
  - Principales problemas
    - Seguridad
    - Carrito de compras *client-side*

Dominios de aplicación

- eBay Shopping API
  - Para buscar productos por palabra clave (en este caso “harry potter”

```
http://open.api.ebay.com/shopping?callname=FindProducts
   &responseencoding=XML
   &appid=YourAppIDHere
   &siteid=0
   &version=525
   &QueryKeywords=harry%20potter
   &AvailableItemsOnly=true
   &MaxEntries=2
```

  - Más información en: http://developer.ebay.com/products/shopping

Dominios de aplicación

- Social Networking
  - Los sitios más importantes proveen interfaces REST
  - OpenSocial
    - API genérica
    - Permite integrar información de diferentes sitios sociales

Dominios de aplicación

- Facebook
  - API híbrida

```
https://api.facebook.com/method/users.getInfo?uids=4
      &fields=name
      &access_token=...
```

- Twitter
  - API puramente REST
  - Para obtener últimos 5 *tweets* públicos

```
http://twitter.com/statuses/public_timeline.xml&count=5
```

Dominios de aplicación

- Sistemas de Información Geográfica
  - Existen formatos de representación específicos para GIS (ej. KML)
  - Aplicaciones con interfaces REST
    - ArcGIS
    - OpenStreetMap

Dominios de aplicación

- ArcGIS
  - Conjunto de aplicaciones para gestionar información geográfica
  - Formato de URL básica

```
http://<host>/<instance>/rest/services/<folderName>
```

  - Ej, para obtener información de un layer

```
http://<catalog-url>/<serviceName>/MapServer/<layerId>
```

Dominios de aplicación

- OpenStreetMap
  - Servicio que provee información geográfica a nivel mundial
  - Actualizado por los usuarios al estilo Wikipedia
  - La unidad básica de información es el *changeset*
  - Para crear un *changeset*, enviar PUT a esta URL

```
http://api.openstreetmap.org/api/0.6/changeset/create
```

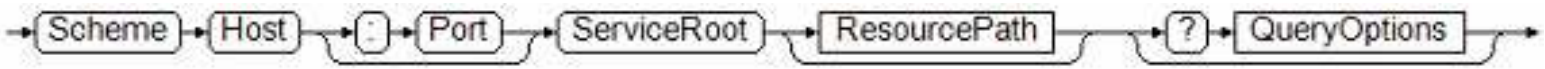
  - En la respuesta se envía el Id del *changeset* creado

Dominios de aplicación

- Bases de Datos
  - OData
    - Protocolo para acceder y actualizar datos en diferentes formatos, vía HTTP
    - Soportado por:
      - MS Sharepoint 2010
      - IBM WebSphere
      - MS SQL Azure
      - Azure Table Storage

Dominios de aplicación

- Formato de URI OData



  - Service Root
    - Documento expuesto por el servicio
  - Resource Path
    - Path al recurso que se busca
  - Query options
    - Opciones de búsqueda

Integración con otras tecnologías

- Mashups
  - Simplifica la obtención de información de fuentes heterogéneas
  - No es necesario definir interfaces particulares
  - Hay pocas iniciativas en este sentido

Integración con otras tecnologías

- ESB
  - Es importante que brinden soporte REST para que más sistemas se puedan comunicar
  - Hoy lo soportan:
    - Mule (ESB opensource)
    - IBM WebSphere ESB

Conclusiones

- Surgimiento con fuerza de REST como alternativa a los Web Services tradicionales

- No solo cambia el formato de la comunicación sino que es una nueva forma de pensar las interfaces