REST: SOA without Contracts?

Stefan Tilkov | innoQ | stefan.tilkov@innoq.com

What is REST?

REST: An Architectural Style

One of a number of "architectural styles"

... described by Roy Fielding in his dissertation

... defined via a set of *constraints* that have to be met

... architectural principles underlying HTTP, defined *a posteriori*

... with the Web as one particular instance

See: http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

REST: The Web Used Correctly

A system or application architecture

... that uses HTTP, URI and other Web standards “correctly”

... is “on” the Web, not tunneled through it

... also called “WOA”, “ROA”, “RESTful HTTP”

REST: XML without SOAP

Send plain XML (w/o a SOAP Envelope) via HTTP

... violating the Web as much as WS-*

... preferably use GET to invoke methods

... or tunnel everything through POST

... commonly called “POX”

RESTful HTTP Explained in 5 Easy Steps

1. Give Every “Thing” an ID

```
http://example.com/customers/1234

http://example.com/orders/2007/10/776654

http://example.com/products/4554

http://example.com/processes/sal-increase-234
```

2. Link Things To Each Other

```
<order self='http://example.com/orders/1234'>
  <amount>23</amount>
  <product ref='http://example.com/products/4554' />
  <customer ref='http://example.com/customers/1234' />
</order>
```

3. Use Standard Methods

GET	Retrieve information, possibly cached
PUT	Update or create with known ID
POST	Create or append sub-resource
DELETE	(Logically) remove

4. Allow for Multiple “Representations”

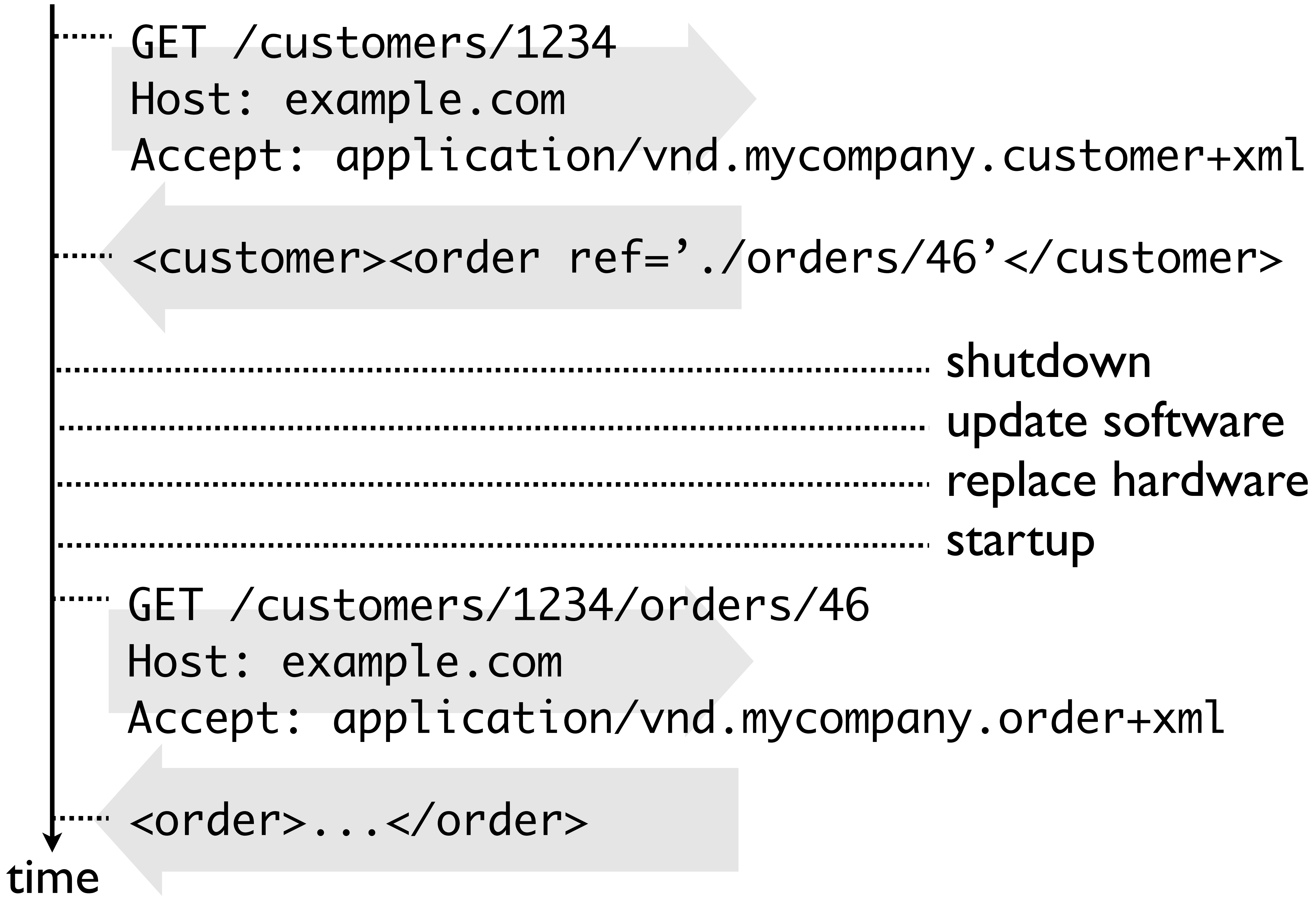
```
GET /customers/1234
Host: example.com
Accept: application/vnd.mycompany.customer+xml
```

```
<customer>...</customer>
```

```
GET /customers/1234
Host: example.com
Accept: text/x-vcard
```

```
begin:vcard
...
end:vcard
```

5. Communicate Statelessly



```
GET /customers/1234
Host: example.com
Accept: application/vnd.mycompany.customer+xml
```

```
<customer><order ref='./orders/46'</customer>
```

```
GET /customers/1234/orders/46
Host: example.com
Accept: application/vnd.mycompany.order+xml
```

```
<order>...</order>
```

What’s cool about REST?

generic

```
interface Resource {
     Resource(URI u)
     Response get()
     Response post(Request r)
     Response put(Request r)
     Response delete()
}
```

Any HTTP client
(Firefox, IE, curl, wget)

Any HTTP server

Caches

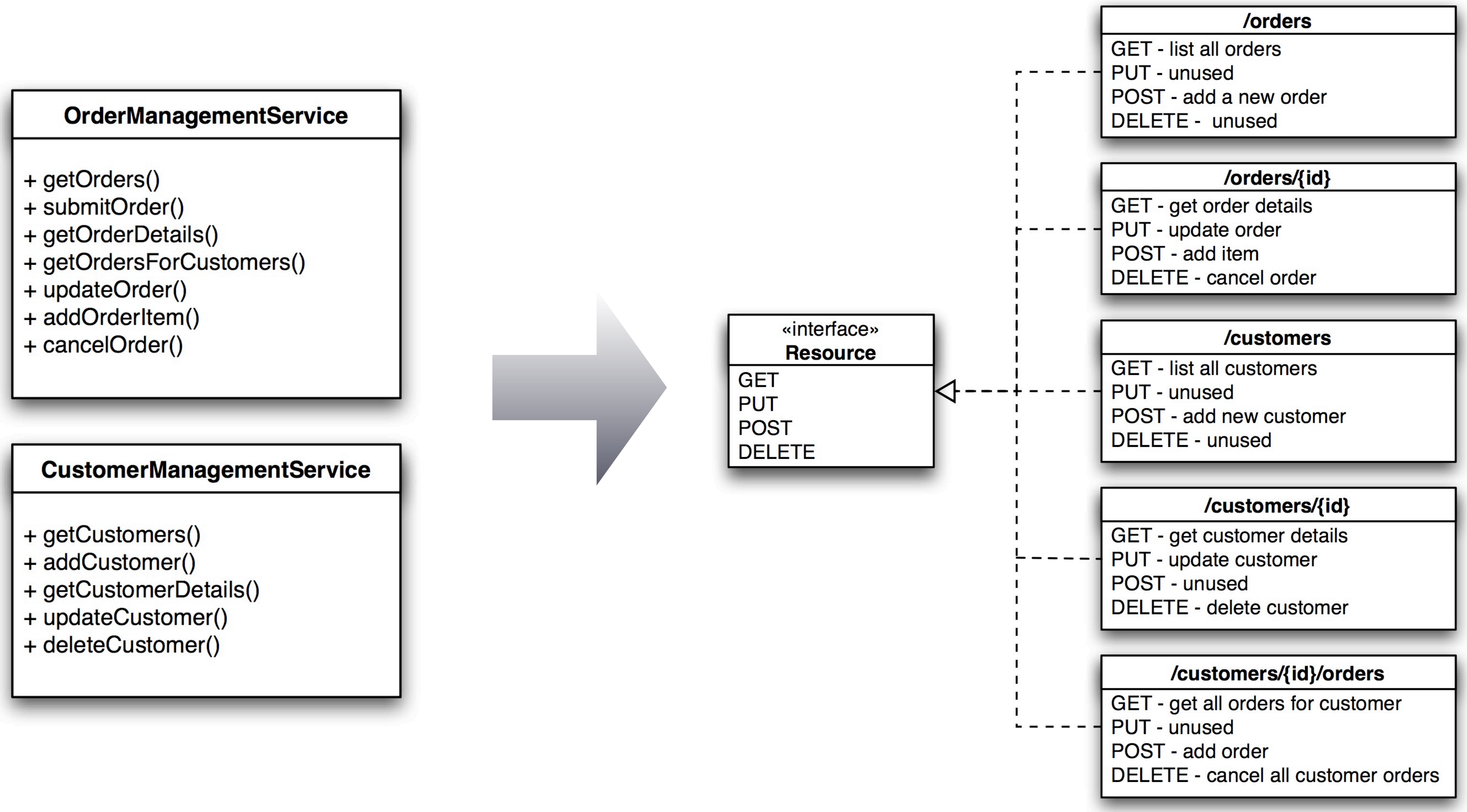
Proxies

Google, Yahoo!, MSN

```
class CustomerCollection : Resource {
     ...
     Response post(Request r) {
          id = createCustomer(r)
          return new Response(201, r)
     }
     ...
}
```

Anything that knows
your app

specific


OrderManagementService
+ getOrders()
+ submitOrder()
+ getOrderDetails()
+ getOrdersForCustomers()
+ updateOrder()
+ addOrderItem()
+ cancelOrder()
CustomerManagementService
+ getCustomers()
+ addCustomer()
+ getCustomerDetails()
+ updateCustomer()
+ deleteCustomer()
«interface»
Resource
GET
PUT
POST
DELETE
/orders
GET - list all orders
PUT - unused
POST - add a new order
DELETE - unused
/orders/{id}
GET - get order details
PUT - update order
POST - add item
DELETE - cancel order
/customers
GET - list all customers
PUT - unused
POST - add new customer
DELETE - unused
/customers/{id}
GET - get customer details
PUT - update customer
POST - unused
DELETE - delete customer
/customers/{id}/orders
GET - get all orders for customer
PUT - unused
POST - add order
DELETE - cancel all customer orders

Mapping Examples

getFreeTimeSlots(Person)	→GET /people/{id}/timeslots?state=free
rejectApplication(Application)	→POST /rejections↵ <application>http://...</application>↵ <reason>Unsuitable for us!</reason>
performTariffCalculation(Data)	→POST /contracts↵ Data ←Location: http://.../contracts/4711 →GET /contracts/4711/rate ←Result
shipOrder(ID)	→PUT /orders/0815↵ <status>shipped</status>
shipOrder(ID) [variation]	→POST /shipments↵ Data ←Location: http://.../shipments/4711

Description

The SOAP/WSDL Problem

Each application is different

Each application requires its own protocol

Need to learn a new API *every single time*

WSDL as formal approach *for syntax only*

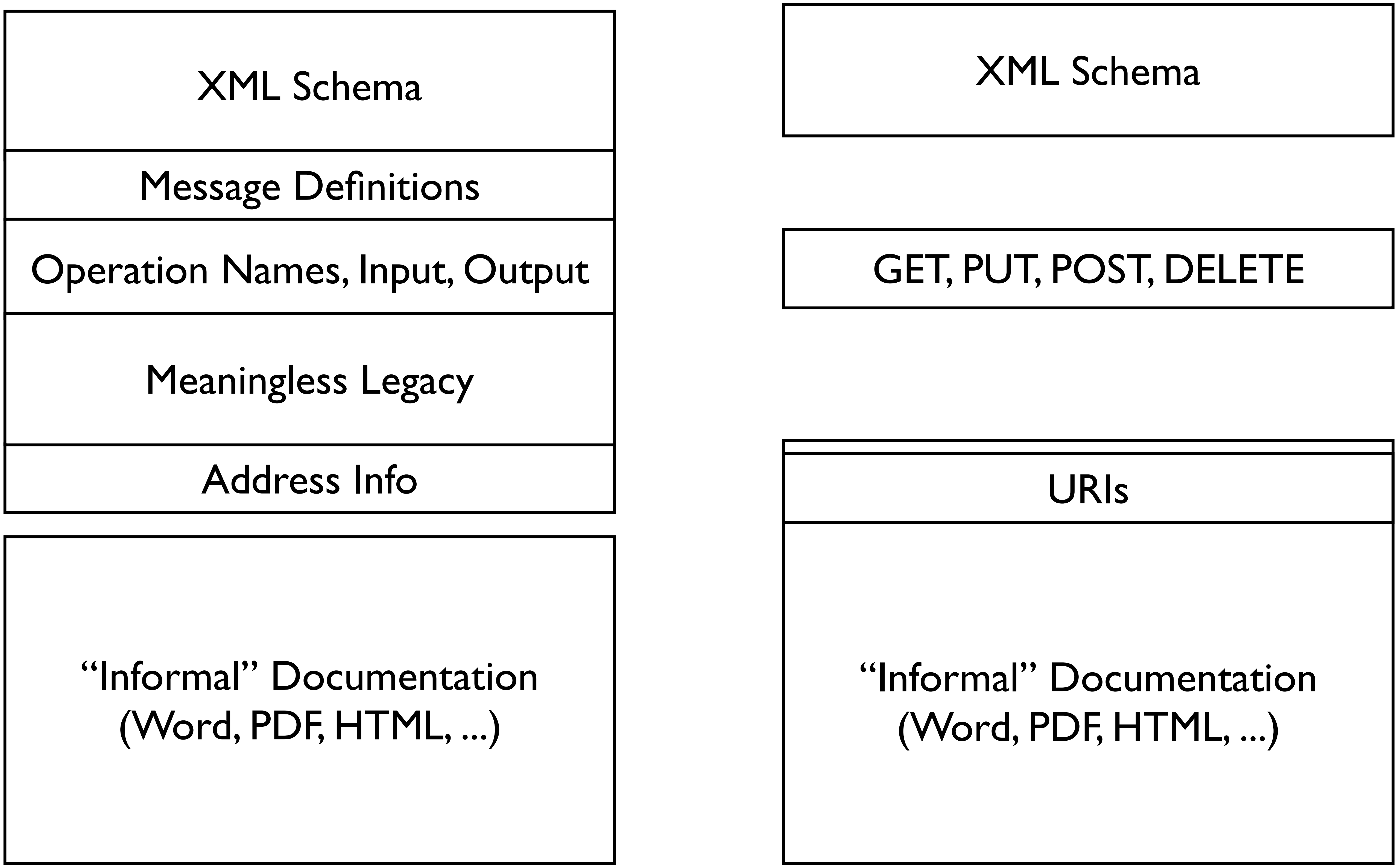
Separation of application and *metadata*

Anatomy of a WSDL File

80%	XML Schema
2%	Message Definitions
5%	Operation Names, Input, Output
10%	Meaningless Legacy
3%	Address Info

SOAP/WSDL

RESTful HTTP


XML Schema
Message Definitions
Operation Names, Input, Output
Meaningless Legacy
Address Info
"Informal" Documentation (Word, PDF, HTML, ...)
XML Schema
GET, PUT, POST, DELETE
URIs
"Informal" Documentation (Word, PDF, HTML, ...)

RESTful HTTP Approach

Data Operations Identity

Data

media types

content negotiation

standard formats

XML Schema & Co.

Operations

minimal set of methods

standardized semantics

uniformity

general applicability

Identity

standardized IDs

cross-application usage

“dereferencability”

ID longevity

“RESTful” Formalisms

WSDL 2.0: Supposedly Usable for REST

- XML-focused and operation-centric
- No content negotiation
- No hypermedia Support

WADL (Web Application Description Language), https://wadl.dev.java.net/

- As RESTful as external metadata can be
- Use cases still doubtful

WADL Example

```xml
<resources base="http://api.search.yahoo.com/NewsSearchService/V1/">
    <resource path="newsSearch">
        <method name="GET" id="search">
            <request>
                <param name="appid" type="xsd:string" style="query" required="true"/>
                <param name="query" type="xsd:string" style="query" required="true"/>
                <param name="type" style="query" default="all">
                    <option value="all"/>
                    <option value="any"/>
                    <option value="phrase"/>
                </param>
                <param name="results" style="query" type="xsd:int" default="10"/>
                <param name="start" style="query" type="xsd:int" default="1"/>
                <param name="sort" style="query" default="rank">
                    <option value="rank"/>
                    <option value="date"/>
                </param>
                <param name="language" style="query" type="xsd:string"/>
            </request>
            <response>
                <representation mediaType="application/xml" element="yn:ResultSet"/>
                <fault status="400" mediaType="application/xml" element="ya:Error"/>
            </response>
        </method>
    </resource>
</resources>
```

Conclusion(s)

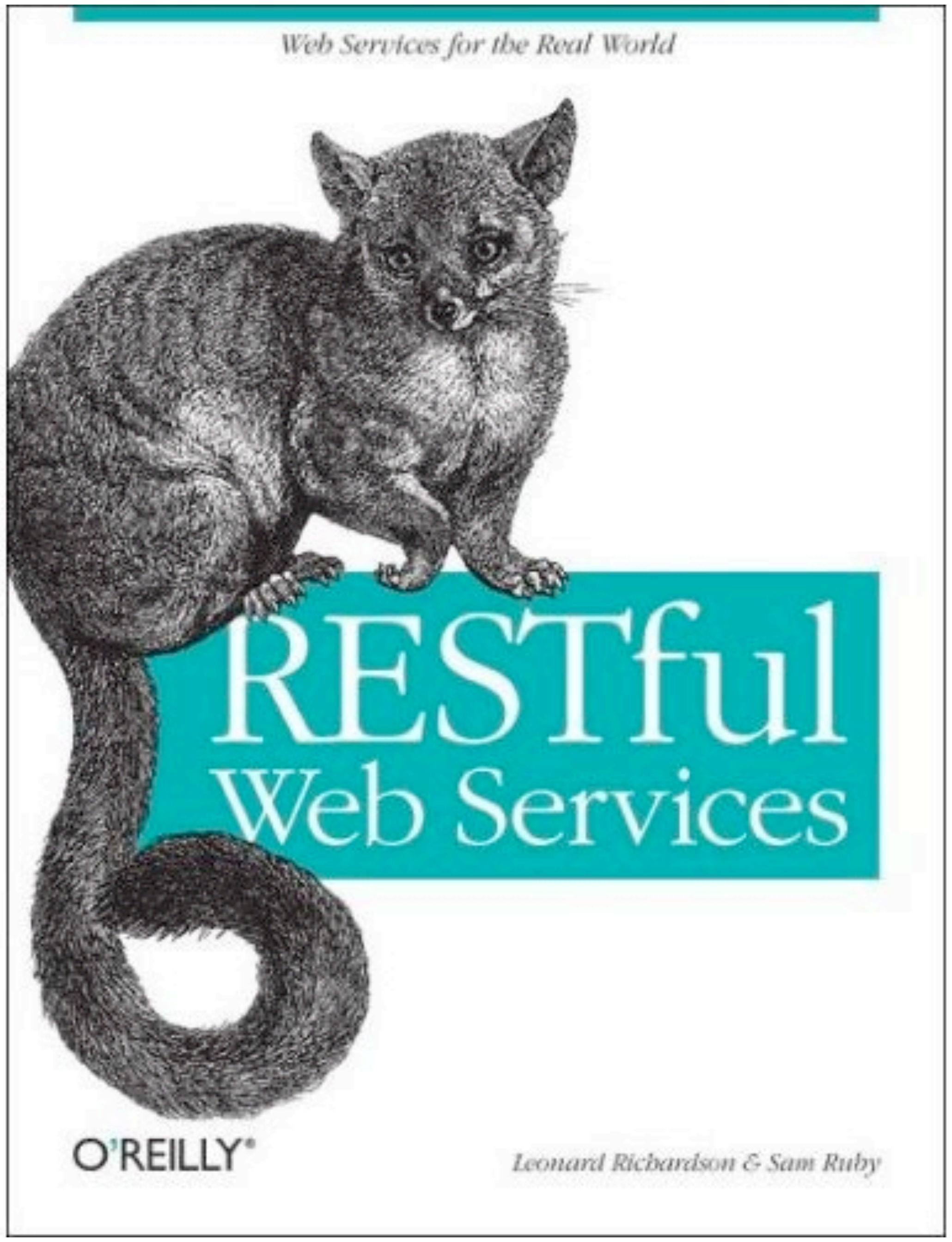
1. **External metadata is a problem, not a solution**

2. Data, operation and identity semantics can be separated

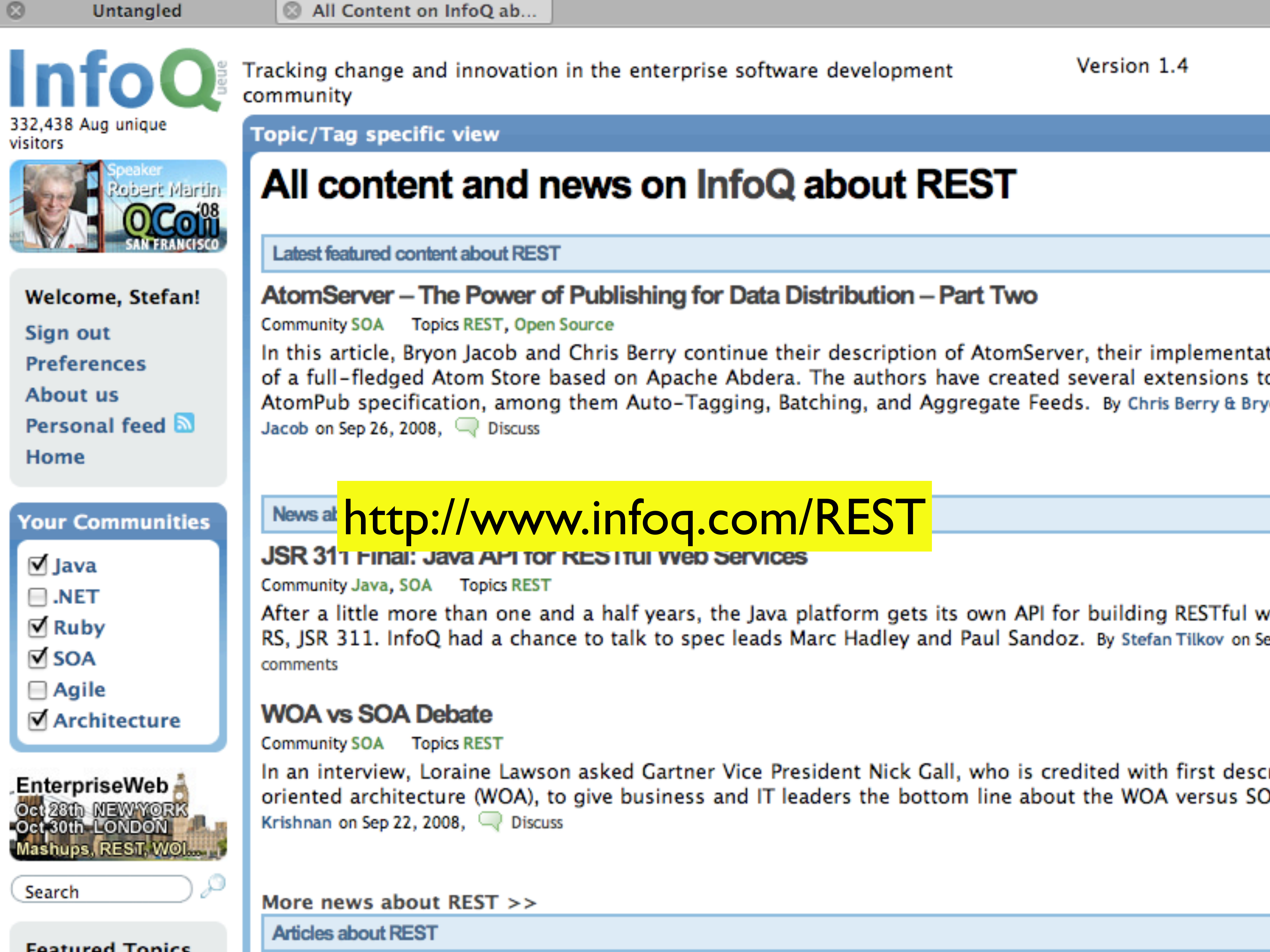
3. The Web is more than you think it is

If You Want to Know More

http://www.innoq.com/resources/REST


Web Services for the Real World
RESTful Web Services
O'REILLY
Leonard Richardson & Sam Ruby

http://www.oreilly.com/catalog/9780596529260/



Untangled | All Content on InfoQ ab...

InfoQ queue

332,438 Aug unique visitors

Tracking change and innovation in the enterprise software development community

Version 1.4

Speaker Robert Martin QCon '08 SAN FRANCISCO

Welcome, Stefan!

Sign out

Preferences

About us

Personal feed

Home

Your Communities

- [✓] Java
- [ ] .NET
- [✓] Ruby
- [✓] SOA
- [ ] Agile
- [✓] Architecture

EnterpriseWeb Oct 28th NEW YORK Oct 30th LONDON Mashups, REST, WOI...

Search

Featured Topics

Topic/Tag specific view

All content and news on InfoQ about REST

Latest featured content about REST

AtomServer – The Power of Publishing for Data Distribution – Part Two

Community SOA   Topics REST, Open Source

In this article, Bryon Jacob and Chris Berry continue their description of AtomServer, their implementat of a full-fledged Atom Store based on Apache Abdera. The authors have created several extensions to AtomPub specification, among them Auto-Tagging, Batching, and Aggregate Feeds. By Chris Berry & Bry Jacob on Sep 26, 2008, Discuss

News ab

http://www.infoq.com/REST

JSR 311 Final: Java API for RESTful Web Services

Community Java, SOA   Topics REST

After a little more than one and a half years, the Java platform gets its own API for building RESTful w RS, JSR 311. InfoQ had a chance to talk to spec leads Marc Hadley and Paul Sandoz. By Stefan Tilkov on Se comments

WOA vs SOA Debate

Community SOA   Topics REST

In an interview, Loraine Lawson asked Gartner Vice President Nick Gall, who is credited with first desc oriented architecture (WOA), to give business and IT leaders the bottom line about the WOA versus SO Krishnan on Sep 22, 2008, Discuss

More news about REST >>

Articles about REST

Thank you!
Any questions?

http://www.innoq.com
http://railsconsulting.de

Stefan Tilkov
http://www.innoq.com/blog/st/

innoQ

Architectural Consulting

SOA	WS-*	REST
MDA	MDSD	MDE
J(2)EE	RoR	.NET

innoQ Deutschland GmbH
Halskestraße 17
D-40880 Ratingen
Phone +49 21 02 77 162-100
info@innoq.com · www.innoq.com

innoQ Schweiz GmbH
Gewerbestrasse 11
CH-6330 Cham
Phone +41 41 743 01 11