

REST for SOA

Stefan Tilkov, innoQ Deutschland GmbH
stefan.tilkov@innoq.com

Contents

An Introduction to REST

Why REST Matters

REST And Web Services

Recommendations

Stefan Tilkov



<http://www.innoQ.com>

stefan.tilkov@innoq.com

<http://www.innoq.com/blog/st/>



<http://www.InfoQ.com>

<http://www.soa-expertenwissen.de>



What is REST?

REpresentational **S**tate **T**ransfer

Described by Roy Fielding in his dissertation

One of a number of “architectural styles”

Architectural principles underlying HTTP, defined *a posteriori*

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

REST Explained in 5 Easy Steps

**0. Prerequisite:
Let's equate "REST" with
"RESTful HTTP usage" ...**

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/customers/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”

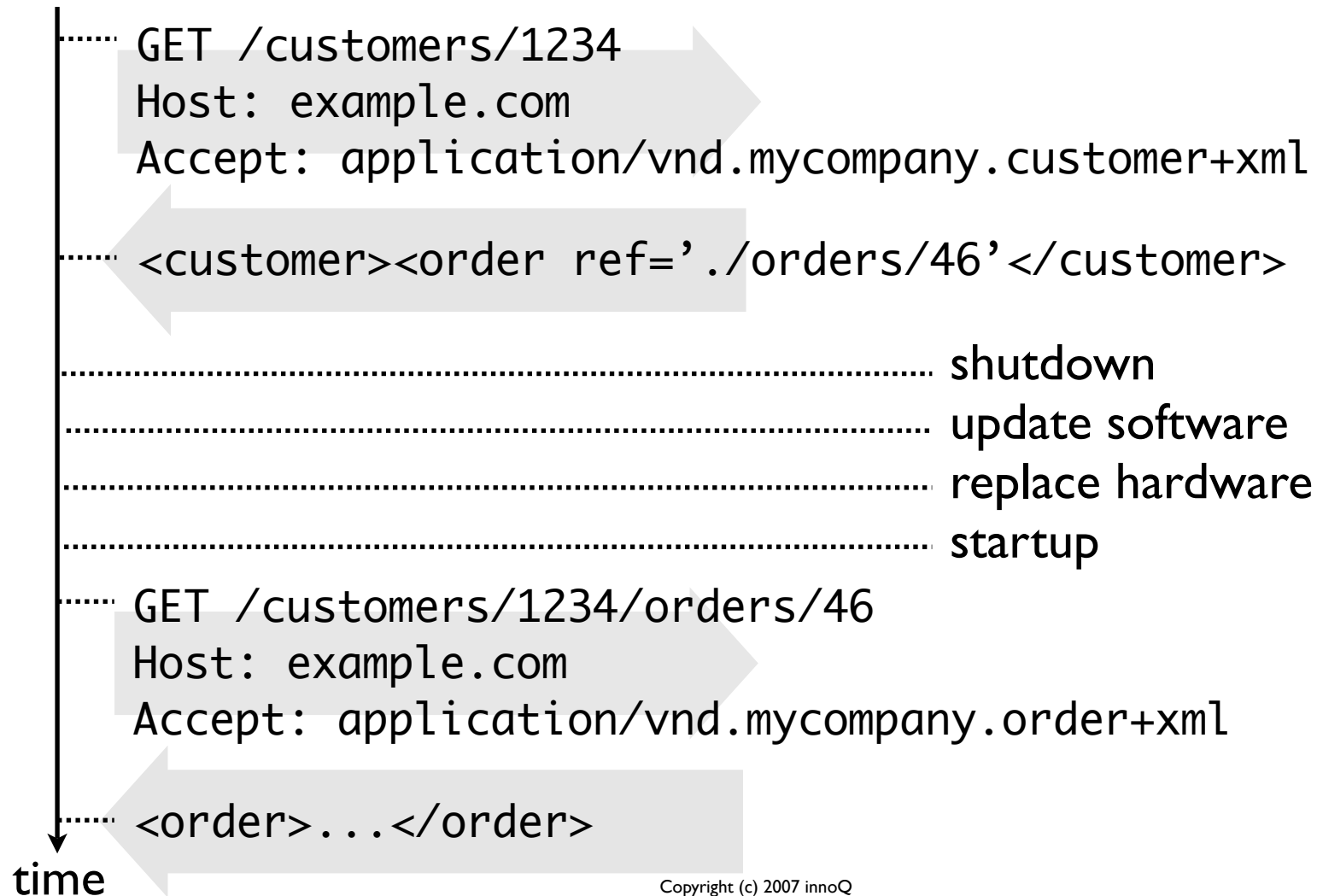
GET /customers/1234
Host: example.com
Accept: application/vnd.mycompany.customer+xml

<customer>...</customer>

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

begin:vcard
...
end:vcard

5. Communicate Statelessly



REST (Pragmatic Version)

- 1 Give everything an ID
- 2 Link things to each other
- 3 Use standard methods
- 4 Allow for multiple representations
- 5 Communicate Statelessly

REST (Academic Version)

- 1 Identifiable resources
- 2 **Hypermedia** as the engine of application state
- 3 Uniform interface
- 4 Resource representations
- 5 Stateless communication

Some HTTP features

Verbs (in order of popularity):

- ▶ GET, POST
- ▶ PUT, DELETE
- ▶ HEAD, OPTIONS, TRACE

Standardized (& meaningful) response codes

Content negotiation

Redirection

Caching (incl. validation/expiry)

Compression

Chunking

Web Services

OrderManagementService

```
+ getOrders()
+ submitOrder()
+ getOrderDetails()
+ getOrdersForCustomers()
+ updateOrder()
+ addOrderItem()
+ cancelOrder()
+ cancelAllOrders()
```

CustomerManagementService

```
+ getCustomers()
+ addCustomer()
+ getCustomerDetails()
+ updateCustomer()
+ deleteCustomer()
+ deleteAllCustomers()
```

A separate interface (façade) for each purpose

As known CORBA, DCOM, RMI/EJB

Often used for SOA (“CORBA w/ angle brackets)

Application-specific protocol

Contribution to the Net's Value

2 URLs

- ▶ <http://example.com/customerservice>
- ▶ <http://example.com/orderservice>

1 method

- ▶ POST

Web Services Issues

Web Services are “Web” in name only

WS-* tends to ignore the web

Abstractions leak, anyway

Protocol independence is a bug, not a feature

Designing a RESTful application

Identify resources & design URIs

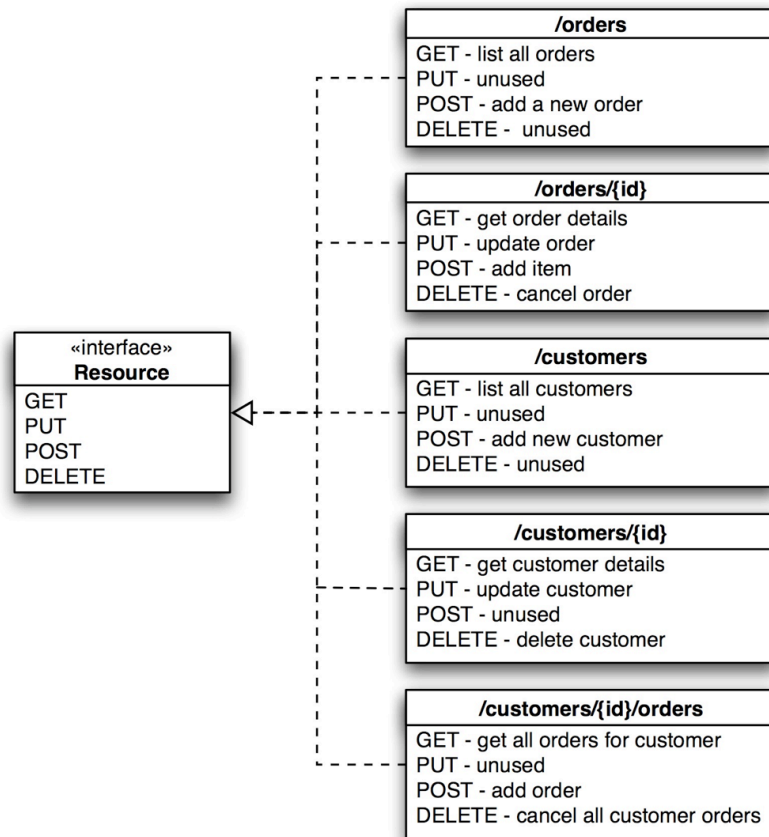
Select formats (or create new ones)

Identify method semantics

Select response codes

See: http://bitworking.org/news/How_to_create_a_REST_Protocol

REST Approach



A single *generic* (uniform) interface for everything

Generic verbs mapped to resource semantics

A standard application protocol (e.g. HTTP)

Contribution to the Net's Value

Millions of URLs

- ▶ every customer
- ▶ every order

4-7 supported methods per resource

- ▶ GET, PUT, POST, DELETE
- ▶ TRACE, OPTIONS, HEAD

Cacheable, addressable, linkable, ...

RESTful HTTP Advantages

Universal support (programming languages, operating systems, servers, ...)

Proven scalability

“Real” web integration for machine-2-machine communication

Support for XML, but also other formats

Why You Should Care

WS-* Roots

The Enterprise

RPC, COM, CORBA, RMI, EJB

Transaction Systems

Controlled Environment

Top-down Approach

REST Roots

The Internet

Text formats

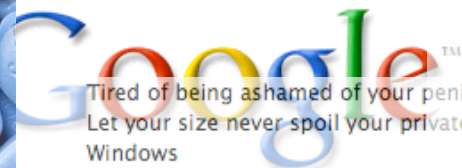
Wire Standards

FTP, POP, SMTP

Bottom-up Approach



- Marguerite E. Lord
- Marguerite J. Lord
- Sales
- Lee J. Lowry
- Lee G. Lowry
- //±q
- Guy C. B...

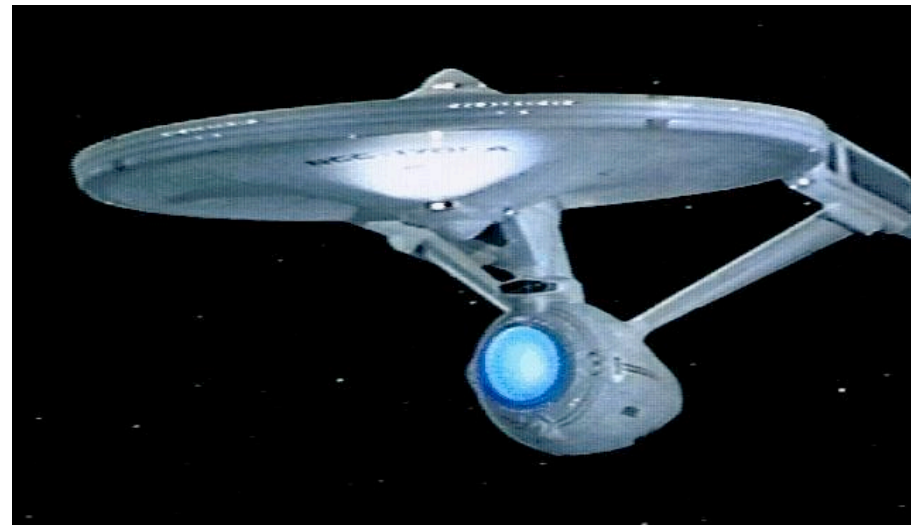


Tired of being ashamed of your penis size? Leave it for l...
 Let your size never spoil your private life!
 Windows
 M5 Off|ce 2007 PRO 79 \$, 5ave 1099.95 Off Retail
 M5 Off|ce 2007 PRO 79 \$, 5ave 1099.95 Off Retail
 Ordinary men have ordinary sex. Megadik will make you ...
 Take care of you and your penis! Enlargement with Mega...
 dookit rcd trichosis
 Prove your manliness! Take MegaDik and be a man!
 Don't be embarrassed every time you get naked! Larger ...
 éöxËç-é±ÆàÀ«U·
 Take MegaDik and enter the reflection of your private life...



Today 1:15 PM
 Today 1:15 PM
 Today 1:18 PM
 Today 1:31 PM
 Today 1:36 PM
 Today 1:50 PM
 Today 1:50 PM
 Today 1:52 PM
 Today 2:16 PM
 Today 2:16 PM
 Today 2:22 PM
 Today 2:25 PM

Internet vs. Enterprise



**What's the difference
between the Internet and a
typical enterprise?**

Internet vs. Enterprise

One is a gigantic, uncontrollable anarchy of heterogeneous systems with varying quality that evolve independently and constantly get connected in new and unexpected ways.

The other is a worldwide, publicly accessible series of interconnected computer networks that transmit data by packet switching using the standard Internet Protocol (IP).

**If web services are
supposed to work on
Internet scale, they should
be inspired by the Web, not
by Distributed Objects**

Quotes

*Frankly, if I were an enterprise architect today, and I were genuinely concerned about development costs, agility, and extensibility, I'd be looking to solve everything I possibly could with dynamic languages and REST, and specifically the HTTP variety of REST. I'd avoid **ESBs and the typical enterprise middleware frameworks unless I had a problem that really required them [...]. I'd also try to totally avoid SOAP and WS-****

Steve Vinoski, formerly IONA

<http://steve.vinoski.net/blog/2007/10/04/the-esb-question/>

“No matter how hard I try, I still think the WS- stack is bloated, opaque, and insanely complex. I think it is going to be hard to understand, hard to implement, hard to interoperate, and hard to secure.”*

Tim Bray, XML Co-inventor

<http://www.tbray.org/ongoing/When/200x/2004/09/18/WS-Oppo>

“Show me the interoperable, full and free implementations of WS- in Python, Perl, Ruby and PHP. You won’t see them, because there’s no intrinsic value in WS-* unless you’re trying to suck money out of your customers. Its complexity serves as a barrier to entry at the same time that it creates ‘value’ that can be sold.”*

**Mark Nottingham, ex BEA, now Yahoo!,
former WS-Addressing WG Chair**

<http://www.mnot.net/blog/2006/05/10/vendors>

*If you're ready for REST I suggest you jump on board right away and get ahead of the curve [...] You'll have to train your developers in REST principles. [...] You definitely need to provide guidance to your people. **What you want to do is work to the point where REST becomes the default for all your distributed applications.***

Anne Thomas Manes, Burton Group

http://searchwebservices.techtarget.com/originalContent/0,289142,sid26_gci1256796,00.html

*“Want to be cool? Learn REST.
Want a career? Learn WS.”*

Steve Jones, Cap Gemini

<http://service-architecture.blogspot.com/2006/11/want-to-be-cool-learn-rest-want-career.html>

Recommendations

1. Be skeptical of the WS-^{*} value-add and pseudo- abstractions

**Protocol
Independence**

Security

Transactions

Reliability

**Application-
Specific Interfaces**

Orchestration

Choreography

2.

**Don't be afraid to make
decisions and depend on
standards**

HTTP, URI, XML, HTML

XML + JMS

SMTP + IMAP

3.

Understand and exploit the Web's architecture to your benefit

**Standardized
Identification**

**Universal
Accessibility**

Scalability

Caching

Hypermedia

Interoperability

UI/API Integration

Slides online → **Stefan Tilkov**
<http://www.innoq.com/blog/st/>



Architectural Consulting

SOA WS-* REST

MDA MDSD MDE

J(2)EE RoR .NET

innoQ Deutschland GmbH
Halskestraße 17
D-40880 Ratingen
Phone +49 2102 77 162-100
info@innoq.com · www.innoq.com

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

Thank you!
Any questions?

<http://www.innoq.com>