REST – das bessere Web-Services-Modell

Stefan Tilkov, innoQ – stefan.tilkov@innoq.com

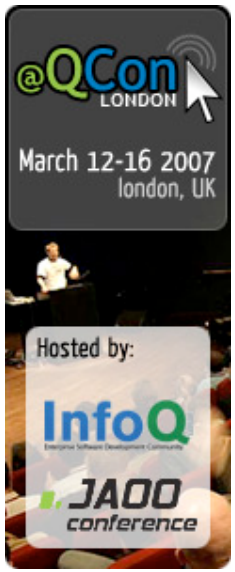
Ziel:

REST erklären und vermitteln, warum die Architektur des WWW für die Lösung Ihrer Probleme geeignet ist

Stefan Tilkov

- Geschäftsführer, innoQ Deutschland GmbH
- C/Windows 1990-1993
- C++/Unix/DCE 1993-1995
- C++/CORBA 1995-1998
- Java/CORBA 1998-1999
- J2EE 1999-2005
- Web Services 2001-2006
- Java EE und REST seit 2006
- Mitglied JSR 311 EG

InfoQ neue
Java mit Integrations-SPEKTRUM
The essence of modern software engineering
OOP 2007
Software meets Business
jax 07



REST als Alternative zu Web- Services

Wenn Sie eine Basis für eine lose gekoppelte, zukunftssichere, verteilte und interoperable Architektur suchen, die auf Standards basiert, sind Web-Services die erste Antwort!

Leider.

Agenda

- Einführung in REST
- REST vs. Web-Services
- Fortgeschrittene Anwendungsfälle
- Zusammenfassung

Agenda

- Einführung in REST
- REST vs. Web-Services
- Fortgeschrittene Anwendungsfälle
- Zusammenfassung

Was ist REST?

- REpresentational State Transfer
- Beschrieben von Roy Fielding in seiner Dissertation
- Einer von mehreren “Architekturstilen”
- Architekturprinzinzien des HTTP-Protokolls, definiert a posteriori

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

REST und HTTP

- REST als abstraktes Konzept könnte mit beliebigen Technologien umgesetzt werden
- Die bekannteste Implementierung ist HTTP
- Im Folgenden vereinfacht REST==HTTP

http://savas.parastatidis.name/2005/03/12/505b74f7-d5d3-4b94-95d4-65129ce2bf2b.aspx

REST Kernprinzipien

- Identifizierbare Ressourcen
- Uniforme Schnittstelle
- Statuslose Kommunikation
- Ressource-Repräsentationen
- Hypermedia

Identifizierbare Ressourcen

- Eine Ressource repräsentiert eine wirkliche oder virtuelle Entität
- Im Web werden Ressourcen durch URIs identifiziert
- Jede URI steigert den “Wert” des Internets
- Was wäre Amazom.com ohne URIs?

Uniforme Schnittstelle

- Ist die URI bekannt, kann man mit einer Ressource über *eine* Schnittstelle interagieren
- Begrenzte Menge an Operationen (*Verben*) in HTTP: GET, PUT, POST, DELETE (+ ein paar mehr)
- Vordefinierte Semantik erlaubt Optimierungen (z.B. Caching)

Ressource-Repräsentationen

- Der Zugriff auf Ressourcen erfolgt immer über Repräsentationen
- Es kann mehrere Repräsentationen geben, z.B. HTML, PDF, XML, JSON
- HTTP unterstützt *content types* und *content negotiation*
- Idealerweise: Standardformate

Statuslose Kommunikation

- Der Server muss keinen kommunikationsbezogenen Status für jeden Client halten
- Massiver Skalierbarkeitsvorteil
- Unterstützt lose Kopplung (keine Bindung an einen spezifischen Server)

Hypermedia

- Statusübergänge über *Links*
- Ermöglicht das Verfolgen von beliebigen Assoziationen
- Links sollten vom Server, nicht vom Client erzeugt werden
- Unterstützt Evolution von Schnittstellen
- Abstrahiert von konkreten Lokationen

Agenda

- Einführung in REST
- REST vs. Web-Services
- Fortgeschrittene Anwendungsfälle
- Zusammenfassung

Gegenbeispiel: (Web) Services

OrderManagementService

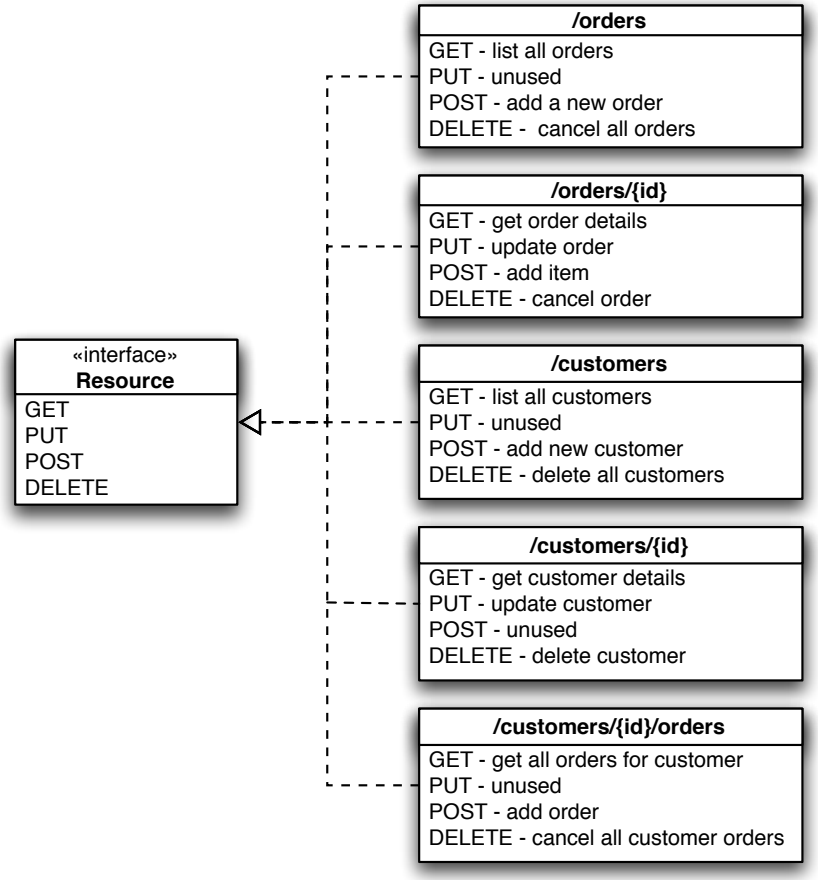
+ getOrders()
+ submitOrder()
+ getOrderDetails()
+ getOrdersForCustomers()
+ updateOrder()
+ addOrderItem()
+ cancelOrder()
+ cancelAllOrders()

CustomerManagementService

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

- Eine neue Schnittstelle (Façade) für jeden Service
- Bekannt aus CORBA, DCOM, RMI/EJB
- Typisch verwendet für SOA ("CORBA mit spitzen Klammern")
- Applications-spezifisches Protokoll

REST-Beispiel



- Einheitliches Interface für alle Ressourcen
- Abbildung generischer Verben auf konkrete Applikationssemantik
- Standard-Applikationsprotokoll

Design einer REST-Applikation

- Ressourcen identifizieren, URIs designen
- Formate auswählen (oder definieren)
- Methodensemantik festlegen
- Response-Codes abbilden

http://bitworking.org/news/How_to_create_a_REST_Protocol

Web Services

2 URLs

- http://example.com/customerservice
- http://example.com/orderservice

1 Methode: POST

REST

1 URL für jede Ressource

- jeder Kunde
- jede Bestellung

4-7 unterstützte Operationen

- GET, PUT, POST, DELETE, ...

Caching, Links, Bookmarks, …

Vorteile des REST/HTTP-Ansatzes

- *Wirklich* universelle Unterstützung (Programmiersprachen, Betriebssysteme, Server, …)
- Vorhandene, hochoptimierte Infrastruktur
- Bewiesene Skalierbarkeit
- Unterstützung für Redirects, Caching, Ressourcen-Identifikation
- Web-Integration für Maschinen-zu-Maschinen-Kommunikation
- Unterstützung für XML, aber auch andere Formate

Die 500.000 €-Frage


RTL
Wie kann man Web-Services so implementieren, dass sie die Vorteile von HTTP nutzen?
A: Microsoft fragen
B: WSDL 2.0/SOAP 1.2
C: WS-Transfer
D: Überhaupt nicht

Welche Features noch mal?

- Einheitliche Identifikation von Ressourcen
- Einheitliches Interface
- Verben (nach Popularität sortiert):
  - GET, POST
  - PUT, DELETE
  - HEAD, OPTIONS, TRACE
- Standardisierte Response-Codes
- Content negotiation
- Redirection
- Caching
- Kompression
- Chunking
- Unterstützung für Nicht-XML-Formate
- Standard-Clients

Stefan Tilkov, innoQ

REST – das bessere Web-Services-Modell

Die 500.000 €-Antwort:

- Wie kann man Web-Services so implementieren, dass sie die Vorteile von HTTP nutzen?
- *Überhaupt nicht*, zumindest nicht ohne entweder gegen das Web oder die Web-Services-Architektur zu verstoßen
- HTTP ist kein “dummes” Transport- sondern ein Applikations-Protokoll

Web-Services und das Web

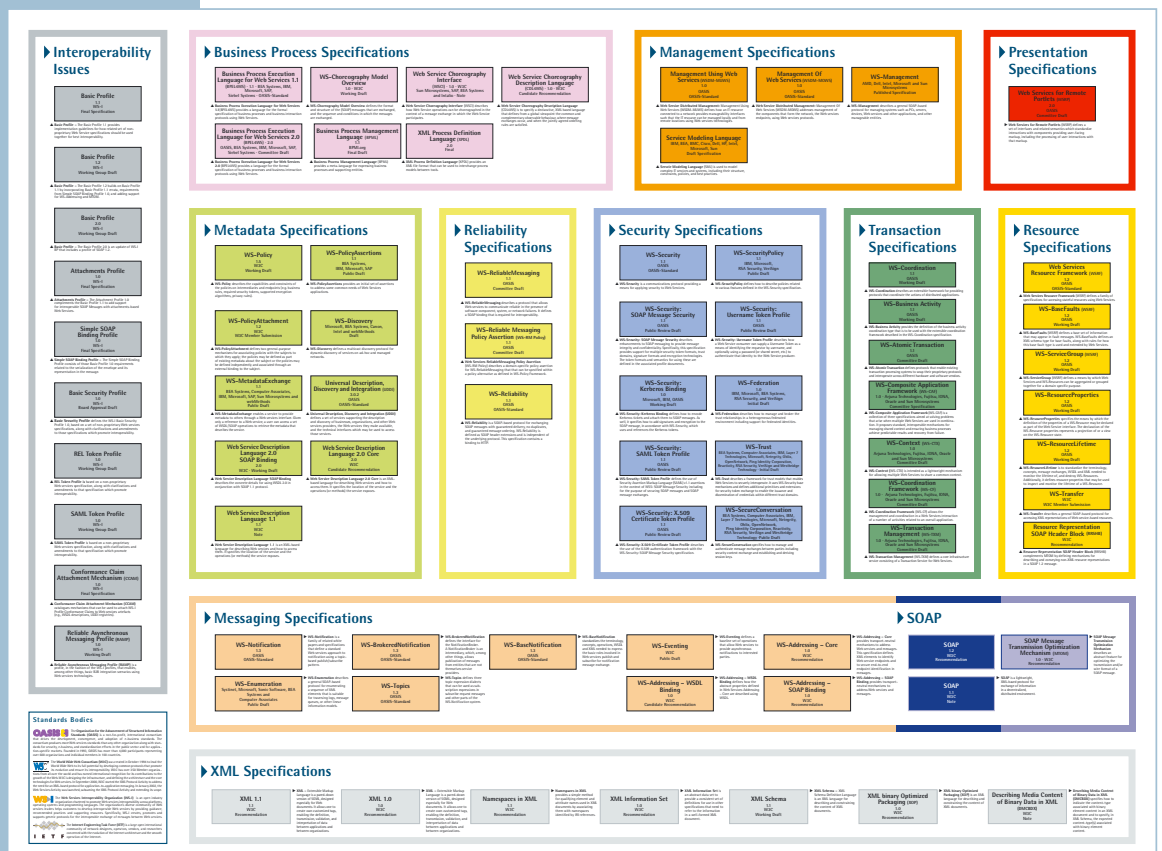
- WSDL/SOAP-basierte Web-Services sind “protokollunabhängig”
- Beispiele:
  - Microsoft unterstützt SOAP über TCP
  - Viele Java-Implementatierungen bieten SOAP über proprietäre JMS-Implementierungen
  - HTTP ist nur ein weiteres “Transportprotokoll”

Das Problem mit Web-Services

Stefan Tilkov, innoQ

REST – das bessere Web-Services-Modell

Web Services Standards Overview



innoQ

innoQ Deutschland GmbH
Halkestraße 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

http://www.innoq.com/resources/ws-standards-poster/

http://www.loudthinking.com/arc/000585.html

"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, (Mit-)Erfinder von XML

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, früher BEA,
ehemaliger Chef der WS-Addressing WG

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

Das Problem mit Web-Services

- Web-Services und WS-* sollen zum neuen, ubiquitären Protokoll-Stack werden
  - aufgesetzt auf einen anderen ubiquitären Protokoll-Stack
- WS-* neigt zum Ignorieren des Webs
- Abstraktionen haben “Löcher”
- Protokollunabhängigkeit ist ein Bug, kein Feature

Das Problem mit Web-Services

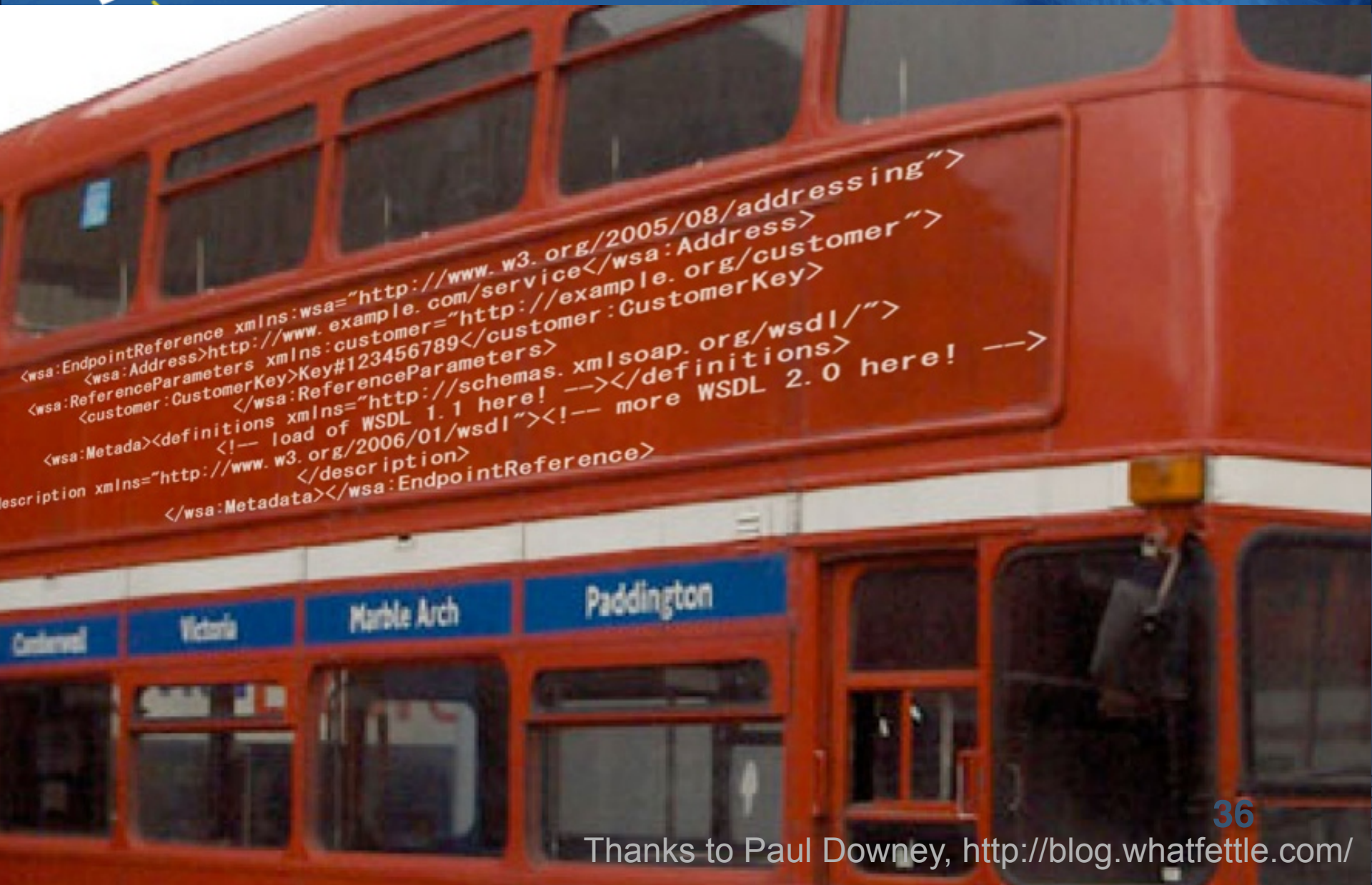
- Das Web und das Internet basieren auf Standard-Protokollen
- Nicht nur HTTP, sondern auch SMTP, FTP, DNS, ...
- Wenn Web-Services auf Internetgrößen-ordnungen skalieren sollen, sollte das Web die Inspiration sein, nicht Distributed Objects oder EAI

Agenda

- Einführung in REST
- REST vs. Web-Services
- Fortgeschrittene Anwendungsfälle
- Zusammenfassung

Asynchrone Kommunikation

- HTTP ist immer Request/Response-orientiert
- Aber: für längere Interaktionen “202 Accepted” + URI für das Ergebnis
- Polling oder Notifikation (bei URI-Übergabe)
- WS-Addressing = URIs in schlecht



Thanks to Paul Downey, http://blog.whatfettle.com/

Reliable Messaging mit HTTP

- Was bedeutet “Reliability”?
- Häufig: Lösung auf Applikationsebene
- Vorschläge:
  - Mark Nottingham, POE (POST Once Exactly)
  - Bill de hÓra, HTTPLR
  - Yaron Goland, SOA-Reliability (“SOA-Rity”)

2-Phase-Commit-Transaktionen

- Der Heilige Gral der Informatik
- In der Praxis viel seltener als angenommen
  - “Es tut weh wenn ich das tue, Herr Doktor.” – ”Dann tun Sie’s nicht!”
- 2PC und lose Kopplung passen nicht sehr gut zueinander
- Kompensierende Transaktionen sind Business Logik
- Leichtgewichtiges Protokoll denkbar – aber bislang nicht spannend genug

Ressourcen-Zugriff

- WS-Transfer

  This specification defines a mechanism for acquiring XML-based representations of entities using the Web service infrastructure. [...] Specifically, it defines two operations for sending and receiving the representation of a given resource and two operations for creating and deleting a resource and its corresponding representation.

- Kommt Ihnen das bekannt vor?
- HTTP-over-SOAP-over-HTTP


HOLD
NEXT
LEVEL
1
GOAL
6
SCORE
506

Discovery - UDDI

420 Seiten zur Pflege von Metadaten

Inquiry
find_binding
find_business
find_relatedBusinesses
find_service
find_tModel
get_bindingDetail
get_businessDetail
get_operationalInfo
get_serviceDetail
get_tModelDetail

Publication
save_binding
save_business
save_service
save_tModel
delete_binding
delete_business
delete_publisherAssertions
delete_service
delete_tModel
add_publisherAssertions
set_publisherAssertions
get_assertionStatusReport
get_publisherAssertions
get_registeredInfo

UDDI (Fortsetzung)

- UDDI wäre mit HTTP direkt sehr viel einfacher umsetzbar
- Es wäre nicht mehr Protokoll-unabhängig – Na und?
- Atom (Syndication Format & Protocol) als ideale Ergänzung

http://www.xml.com/pub/a/ws/2002/02/06/rest.html?page=2

Servicebeschreibung

- Mainstream Web-Services: WSDL 1.1 + XSD
- WSDL 2.0 (Status: Last call WD) erlaubt auch Beschreibung von REST-Services
- Alternativen:
  - WADL (Web Application Description Language)
  - XSD/RELAX NG + HTML/RTF/PDF

REST im Mainstream

- Implizit in allen HTTP-APIs
- REST-Unterstützung in
  - Microsoft WCF
  - Axis2
  - XFire
- *simply_restful* in Ruby on Rails 1.2.x
- JAX-RS: Java API for RESTful Web Services (JSR 311)

Attachments (Binärdaten)

- In der WS-*-Welt:
  - SOAP mit Attachments (MIME)
  - DIME (unterstützt von Microsoft)
  - XOP/MTOM
- In einer “RESTful” HTTP-Applikation:
  - Per Link oder
  - als alternative Repräsentation

Sicherheit

- Web Services Security (WSS) ist nachrichtenbasiert
- HTTP verwendet
  - Transportsicherheit (SSL/TLS)
  - *Basic* und *Digest Authentication*
  - Zugriffskontrolle auf Basis von Ressourcen und Methoden
- WSS-Konzepte wären eine gute Ergänzung

Agenda

- Einführung in REST
- REST vs. Web-Services
- Fortgeschrittene Anwendungsfälle
- Zusammenfassung

Fazit

- REST ist die Architektur der erfolgreichsten verteilten Systems der Welt
- Web Services haben mit dem Web wenig zu tun
- HTTP ist kein Transportprotokoll
- “Einfach nur HTTP” ist oft die beste Option

If You Only Remember One Thing …

HTTP is Good Enough.

Vielen Dank!
Noch Fragen?

Stefan Tilkov
http://www.innoq.com/blog/st/
stefan.tilkov@innoq.com

innoQ

innoQ Deutschland GmbH
Halskestraße 17
D-40880 Ratingen
Phone +49 21 02 77 162-100

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

info@innoq.com · www.innoq.com