ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

Εισαγωγή στα Δίκτυα Υπηρεσιών

Διάλεξη 9η: RESTful Services

Χρήστος Νικολάου
Τμήμα Επιστήμης Υπολογιστών

Introduction to Service Networks

RESTful Services

Christos Nikolaou
Mariana Karmazi
Transformation Services Laboratory
CSD/UoC

CS452 Fall 2010-2011

REpresentational State Transfer (REST) Web Services

REST unlike SOAP, WSDL, UDDI, is **not** a spec, it is a set of principles and concepts (similar to SOA, ESB, ...)

REST is the brainchild of Roy T. Fielding, cofounder of the Apache HTTP server project and coauthor of the HTTP and URI standards. In 2000, he wrote a doctoral dissertation for the University of California at Irvine called *"Architectural Styles and the Design of Network-based Software Architectures"*, wherein he defined REST.

REST doesn't build on the principles of the Web—the Web was built based on RESTful principles. The idea of REST is essentially a reverse-engineering of how the Web works. HTTP itself, and URIs themselves, are written with REST principles.

REST has a growing popularity, and it is often radically misunderstood. It seems that the popular imagination has taken hold of the idea of REST as any web services that are not SOAP with WS-*. Such designs are popular and abundant, and may have many laudable qualities, but they are not an embodiment of REST just because they aren't SOAP.

Source: Eben Hewitt, Java SOA Cookbook, May 2009

What REST really is

Industry analysts have recently coined the term WOA, or Web Oriented Architecture: SOA-lite, or a service architecture that uses XML, JSON (JavaScript Object Notation), and HTTP. The term is generally intended to imply two things: an architecture that

1) does not use WS-* specifications and

2) is an application of RESTful principles that is much looser than Fielding's dissertation actually allows (you can use cookies, etc.).

Principles of REST: Statelessness

- REST Services are stateless:"each request from client to server must contain all of the information necessary to understand the request, and cannot take advantage of any stored context on the server" Fielding, (pages 78–79).
- Therefore, clients must store any state necessary.
- Server *sessions* should *not* be used, since everything that you need to process a request should be contained in that request.
- Cookies violate REST, according to Fielding, since they typically store information that encompasses a user's interaction with an entire site and not just the current request, which violates encapsulation. Furthermore, cookies pass data without defining the semantics that define that data, thereby raising concerns regarding both security and privacy.

Principles of REST: Uniform Interface

There is no WSDL in REST. This constraint on the architectural style of REST is gen erally understood as the interface provided by the standard HTTP methods (PUT, GET, POST, DELETE, etc.), but Fielding doesn’t say that. He actually stresses protocol in-dependence, despite the popular reliance on HTTP.

Principles of REST: Resources are manipulated through Representations

The components in the system exchange data (usually XML documents) that represents the resource. For example, if you wanted to update customer information, you would post the XML document representing that resource to its URI. A representation is simply a sequence of bytes and metadata in the form of name/value pairs that describe the resource.

Resources have multiple representations. The resource is not the HTML page that might be returned to a browser when you request it—that is simply one out of many possible representations of the resource. You could return a variety of formats repre- senting a given resource, as stated in section 5.2.1 of Fielding's dissertation:

- XML
- JSON
- XHTML
- JPEG image

Principles of REST: Messages are self-describing

No out-of-band negotiation can be required to determine how to communicate with the service.

Principles of REST: RESTful Architectures are built with Resources

RESTful architectures are built through *resources,* each of which has its own unique URI.

A **resource** is an item that "might be the target of an author's hypertext reference" (section 5.2.1.1). The URI serves as the ID of the resource. Any information item that can be named can be a resource. This could be a stock price at a given point in time, a purchase order, the current weather in Scottsdale, and so on. Here are some examples:

- http://MyBusiness.com/customers
- http://MyBusiness.com/customers/1234
- http://MyBusiness.com/orders/456/customer

RESTful identifiers may be long-lived and stable. Because of the use of URIs, everything that matters to users in REST can be bookmarked, cut and pasted, or cataloged with RDF (Resource Description Framework) in a metadata model. Every unit of information carries its address, and is wholly independent of the underlying framework or implementation producing the API. Clients simply act on the **representations** they receive.

Principles of REST: Hypermedia is the Engine of Application State

Hypertext is text that is interactive and nonlinear, that branches and gives the reader a set of choices.

Fielding's site defines hypertext as "the simultaneous presentation of information and controls such that the information becomes the affordance through which the user (or automaton) obtains choices and selects actions."

This means that every document returned by the server will include all the URIs to any next step. That is, all possible application states that the user can transition to from the current state are represented as resource URIs (hypermedia links). Application state is driven (transitioned to a next state) by selecting and following a URI.

On his website, Fielding recently restated the case that hypertext is a constraint: "if the engine of application state (and hence the API) is not being driven by hypertext, then it cannot be RESTful and cannot be a REST API. Period. Is there some broken manual somewhere that needs to be fixed?" (http://roy.gbiv.com/untangled/2008/rest-apis-must -be-hypertext-driven)

Hypermedia is the Engine...Cont.

REST really takes issue with cookies because they carry state and are intended to be sent to the server in future requests. Because some state is set in the cookie, the application avoids representing all possible next-state transition possibilities directly in the hypertext.

This constraint is frequently ignored, however, in “RESTful” framework implementations in the real world, not the least of which is JSR 311, which includes the @CookieParam annotation to allow a provider to extract information from cookies, as well as access to the Cookie class. So there are competing degrees of “purity” among REST practitioners.

It’s fine to use cookies if you need to. There are loads of popular websites that use cookies. Just don’t call that application RESTful.

So in REST, how do you keep state for something like a shopping cart? Fielding’s suggestion, instead of using cookies to identify a set of cart products within a server-side database, is to define the semantics of the products within hypermedia data formats, so that the user can store selected items client-side and use a URI to check out. To restate the point, the URI serves as the ID of the resource. Any information item that can be named can be a resource. “Bob’s shopping cart on Tuesday at 9 a.m.” is a valid resource.

Advantages of REST

Benefits directly inherited from building on the architecture of the Web in its static form, on the server side:

- ability to scale horizontally
- an easy and clear mechanism for caching,
- a simple failover strategy.

On the client side:

- ability to cache and bookmark representations,
- the flexibility to choose data formats that are most appropriate for the use case.

The REST criticism of the WS* stack (see also “Semantic Web Services”, Fensel et al.)

- The technical landscape around Web services is scattered—spanning according to surveys more than 80 Web services standards, technologies and specifications
- This diversity is related to a high learning curve, which is required to get familiar with the overall area, identify the most important developments, and stay up-to-date with the latest advances.
- The WS-* technology stack does not have much in common with the Web, despite the fact that their very name suggests something else. They do not necessarily follow the same design principles and do not rely on the same core formats and protocols that have lead to the undeniable success of the Web as a global medium for communication.

The REST criticism of the WS* stack

- In the Web, an information provider makes information available by publishing it online, and this information is then accessible to the intended readers.
- In contrast, Web services are frequently geared towards targeted messaging, with obvious consequences in terms of scalability.
- When sending and receiving SOAP messages, the content being exchanged is hidden in the body of the message, and not addressed as an explicit Web resource by its own URI.

The REST criticism of the WS* stack

- Consequently, all Web machinery involving caching and security is disabled since its use would require parsing and understanding of all possible XML dialects that could be used to write a SOAP message.
  - referring to content via an explicit URI in an HTTP request would allow the content of a message to be treated just as any other Web resource.
- the Web service technology stack is oriented towards modeling stateful resources, in contradiction to the REST (REpresentational State Transfer) architectural principles underlying the World Wide Web.
- due to its stateless design, application integration and servers for this architecture are easy to build. Every HTTP request for a URI should retrieve the same content independent of what has happened before in other sessions, or in the history of the current session. This allows the use of thin servers that do not need to store, manage and retrieve the earlier session history to process the current session.
- When a stateful conversation is required, this should be explicitly modeled by different URIs.
- Applying these principles to Web services-based communication means that there should not be a single URI for a Web service, or hidden ways to model and exchange state information; instead, each potential state of a Web service should be explicitly addressable by a different URI.

WADL for RESTful services: an example – yahoo news search

https://github.com/blackwinter/wadl/blob/master/example/YahooSearch.wadl

Τέλος Ενότητας

Χρηματοδότηση

•Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.

•Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.

•Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Σημειώματα

Σημείωμα αδειοδότησης

- Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά Δημιουργού - Μη Εμπορική Χρήση - Παρόμοια Διανομή 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».

[1] http://creativecommons.org/licenses/by-nc-nd/4.0/

•Ως **Μη Εμπορική** ορίζεται η χρήση:

–που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο

–που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο

–που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

•Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

.

Σημείωμα Αναφοράς