



Designing RESTful Services

The REST vs. Web Services debate may never come to an end. But for those individuals who are tired of debating the academics and are ready to move to REST, 'Designing RESTful Services' is the answer. Whether you're an old hand at Web Services or a novice approaching services for the first time, this course provides a practical, hands-on approach to understanding both basic and advanced concepts.

The course provides attendees with a mix of theory, practical guidance and concrete examples. Attendees are also walked through a series of labs to reinforce the concepts and techniques.

Objectives:

- Understand the motivation for REST
- Understand the fundamental Rest-Oriented Architecture
- Understand the steps to analyzing and designing RESTful systems
- Understand how to overcome common obstacles in REST design
- Understand issues related testing services
- Understand the issues of versioning and preparing a service for production

Audience:

This course is designed for software designers who need to define the actual services. Architects and programmers may find this course valuable to understand the upstream & downstream deliverables.

Prerequisites:

- Attendees should have a software development background
- Familiarity with XML is recommended

Duration:

2 days

Outline:

1. Thinking in Terms of Clients & Services

- Overview: What are clients?
- Overview: Elements of Service
- Describing Software as a set of Services
- The client/service relationship
- Intermediaries
- Service: The new unit of work
- A look at upstream analysis (SOA Analysis)
- Eliciting the current and future needs of a client
- Failed encapsulation
- Determining where the logic goes: client/intermediary/service
- Interfaces vs. Implementation
- Skeletons: Stringing services into applications

2. Introduction to REST

- Why the Web was Successful
- The Role of REST in the Web
- The Origins of REST
- The Benefits of REST
- Ubiquity Through HTTP and XML
- Competing Architectures

3. Resource Oriented Architecture

- Introducing Resources
- Linking with URI's
- Everything is Addressable
- Keeping Services Stateless
- Uniform Interfaces
- Representing the Information

4. Understanding HTTP

- An Overview of HTTP
- Get / PUT / DELETE
- POST
- Safety & Idempotence
- Using HTTP HEAD and OPTIONS
- HTTP Response Codes

5. Hypermedia as Application State

- Resource vs. Application State
- Thinking in Terms of State Machines
- Intermediaries and State Propagation
- State Consistency

6. Introducing Atom

- The Role of Atom
- The Atom Syndication Format
- The Atom Publishing Protocol
- Atom Extensions: Search & Page Feed
- Scenarios where Atom is essential

7. Resource Analysis & Design

- Distinguishing Nouns and Verbs
- Analyzing the Domain
- Identifying the Resources
- Identifying Lists / Feeds
- Creating URI's and Taxonomies
- Associating Resources with Verbs
- Determining the Representations
- Linking & Associating Resources
- Resource Refactoring
- Specifying constraints & validations
- Dealing with errors & exceptions

8. Addressing Common Concerns

- Making RESTful Services Scale
- Fatty vs. Chatty
- Time bound coupling
- Ensuring Reliability
- Implementing Security
 - Authentication
 - Authorization
 - Encryption
- Implementing Transactional Integrity
- Making High Performance Services
 - Compressing the Payload
 - Caching

9. REST Frameworks

- Java & JAX-RS
- Ruby & RoR
- .Net and WCF

10. Creating Service Test Cases

- Overview: Test Driven SOA
- Planning the tests
- Designing the tests
- Implementing test cases
- Creating a testing environment
 - Building mock services
- Coordination with Quality Assurance

11. Client Interactions with RESTful Services

- Keeping State on the Client
- Thin Client Interactions
- Warnings on Cookies
- AJAX Interactions
- SilverLight & Adobe Interactions

12. Service Refactoring & Versioning

- Iterating on a service design
- Client requirements expressed as Venn diagrams
- Refactoring services
- Refactoring operation functionality to increase reusability
- Supporting extensibility
- Breaking backward compatibility
- Communicating new versions
- Working with configuration mgmt.

13. Documentation & Publishing

- Overview: Who needs Service documentation?
- Documenting RESTful Services
- Publishing the documentation
 - Web pages and repositories