Replacing Legacy Web Services with RESTful Services

Charles Engelke
Craig Fitzgerald
Info Tech, Inc.

Background

- Bid Express® (www.bidx.com) is a system for transportation construction bidding
  - 13 years in operation
  - Very widely used
- Signed sealed bids submitted via web services
  - RPC over HTTP
- Replacing the web services with RESTful ones
  - Covering new use cases
  - Improved interoperability

Current Operation

- Request via HTTP POST
  - Body contains form data
  - One element specifies function, others are input parameters
- Response is plain text
  - Always status 200 OK, content says success or failure
  - Response data encoded as needed as text
- No user authentication
  - Request body is digitally signed

Existing Web Service Issues

- The web services aren't complete solution
  - Some steps done interactively, elsewhere
- Specialized business rules on server
  - Reducing ability to extend to other uses
- Services partitioned by user
  - Instead of data or function
- No advantage taken of HTTP features
  - Leading to proprietary re-invented "wheels"
- But excellent track record over a long period

Why Move to RESTful?

- New requirements require changes anyway
  - Cryptographic protocols, multi-part bids, new client types
- New use cases with similarities to bidding
  - An architecture that drew clear line between moving resources and processing them would help
- RESTful architecture seemed good fit
  - Bidders, agencies, bids and keys map naturally to resources
  - HTTP standards map well to operations
  - Good tools available for RESTful services

Bid Submission Use Case

- Traditional Story
  - A transportation agency publicly announces requests for bids.
  - Bidders prepare their bids and deliver them to the agency in a sealed envelope prior to a published deadline.
  - Shortly after the deadline, the agency opens the envelopes in public and reads and processes the bids.
- Electronic Version
  - Sealing data via public-key cryptography
  - Signing via digital signatures

Similar Use Cases to Support

- Certified Payrolls
  - Contractors must list all employees on project and certify (via signature) all payments to them.
  - Contains highly confidential personal information.
- Subcontract Requests
  - Prime contractor describes work to be subcontracted, signs and delivers to agency. Agency signs and returns if approved.
- Design Estimates
  - Consultant prepares list of work items and quantities and estimates their cost. Signs and delivers to agency.
  - Estimates are confidential until after bidding.

Comparing Use Cases

Similarities

- One party requests data, another provides it
- Only the creator and the recipient can see data
- Sender must be authenticated
- Data must be signed
- Data must often be sealed

Differences

- Business rules
- Timing (deadlines)
- Data formats
- Creation and processing data

New Service Overview

- Address only the commonalities on the server
  - Leave differences to the clients
- Resources are
  - Things (atomic resources with meaning to clients)
  - Places (collections of Things)
  - Spaces (collections of Places)
- Resource contents in HTTP bodies
- Metadata placed in HTTP headers
- Actions map to standard HTTP verbs
- All requests authenticated

Special Rules

- The creator of a resource “owns” it
  - Many operations are only allowed for the owner or the resource, or the collection containing it
- Places have “policies”
  - Publish
    - Anyone can see the Place and Things within
    - Only the Place owner can add Things
  - Deposit
    - Anyone can add Things
    - Only Thing owner or Place owner can see Things

Additional Optional Deposit Rules

- Signature
  - Can only add Things that are digitally signed
  - Root certificate URIs a property of the Place
- Encryption
  - Should only add Things properly encrypted
  - Recipient key URI a property of the Place
- Deadline
  - Can only add Things prior to deadline
  - Place owner can only see Things after deadline

Sealed Bidding with Elephant

- Agency creates Deposit Place
  - Specifies deadline, signing rules, sealing rules
  - Publishes Place URI to bidders
- Bidder prepares bid document
- Bidder submits bid
  - POST to Deposit Place
  - Must be properly signed and sealed
- Bidder can create, view, update, delete prior to deadline
- Agency can retrieve, decrypt, analyze only after deadline

Development Experience

- Interoperability was a major goal
  - Created servers on Windows and Linux with Apache/Perl/CGI and Ruby on Rails
    - Preliminary version in Python on Google AppEngine
  - Clients for Linux command line, Windows command line, Windows GUI, web browsers
    - C# in .NET, JavaScript on IE 7 and 8 with CAPICOM, JavaScript on Firefox and Chrome, Perl with OpenSSL, Ruby with OpenSSL, curl with OpenSSL
- Every combination of client and server tried
  - Some did not support cryptographic rules
  - Had to work around issues with PUT and DELETE

Results

- Elephant server in production use
- Initial uses
  - Contractor to Agency secure data communications
  - Bid submission for non-transportation agencies
- Future uses
  - All bid submission
  - Secure point to point connections

What We Hope to Change in Next Version

- Do even less
  - Specification addresses complexities that turned out to be unnecessary (e.g., recursive search)
- More links
  - Now often use names of resources, not links with full URIs, requiring more knowledge in clients
  - Move from Richardson Maturity Model level 2 to 3
- Pay more attention to Ajax clients
  - Web browsers have file handling limitations needing special help (e.g. Content-Disposition, OPTIONS)