
PRESENTED BY RED HAT

# LEARN. NETWORK.
# EXPERIENCE OPEN SOURCE.

www.theredhatsummit.com

# Why RESTful Design for Cloud is Best

Galder Zamarreño
Senior Software Engineer, Red Hat
25th June 2010



# Who is Galder?

- Core R&D engineer on Infinispan and JBoss Cache
- Contributor and committer on JBoss AS, Hibernate, JGroups, JBoss Portal...etc



# Agenda

- Brief introduction to RESTful principles
- Cloud APIs and REST
- Cloud storage with Infinispan REST module
- Applying RESTful principles to other APIs



# What is REST?

- REpresentational State Transfer (Roy Fielding)
- An architectural style
- Part of the web from day 1 (sort of...)
  - Evolved with HTTP 1.1



# What it is not

- A protocol
- An interface
- An API
- A standard
- A drop in replacement for SOAP (or RPC)



# What is right with REST

- Fixed well defined verbs
  - GET, PUT, POST, DELETE, HEAD...
- Focus on resources (not services)
  - Makes “service implementors” work harder
- Requests are stateless
- Hypermedia As The Engine of Application State (HATEOAS)


# Cloud APIs and REST

- 2 categories for Cloud APIs with different requirements
  - IT/Cloud resource management
    - Non-REST APIs: SNMP, WBEM, WMI, JMX/RMI, OSGI...etc
  - Cloud storage
    - Non-REST APIs: NFS, CIFS, WebDAV...etc
- REST can provide a consistent set of principles


# REST for resource management APIs

- Short term:
  - REST reduces invalid state transition calls
  - REST is to IT management protocol design what classical music training is to a rock musician
- Long term:
  - REST enables IT management console “in the cloud”
- But above all:
  - The API doesn't constrain the design of the URI space



# What REST can't do for resource management

- Querying: "*Get all JBoss AS instances running on Linux that don't have patch JBAS-1234 applied*"
  - Links are part of resource representation, not resources of their own
- Retrieving resource configuration change history
  - Can you guarantee that URIs will remain unchanged?
- These problems beg for an RPC pattern



# Who is using REST for resource management?

- Sun (Oracle?) Cloud API
  - Consistent many-as-one and direct resource access
  - JSON only – ready for versioned/complex models?
- Rackspace
  - Multiple representation – JSON/XML...etc
- EC2
  - No REST; Plain HTTP or SOAP



# REST for cloud storage APIs

- REST promotes a “stateless” architecture
  - No need for maintaining sessions, easier to scale up
- Improved network efficiency thanks to caching
- Internet is highly latent
  - NFS or CIFS only valid in local network access



# REST for cloud storage APIs

- WebDAV is useful over an internet connection
  - … but limited to traditional file operations
  - i.e. store, retrieve...etc
- REST offers a solution...
  - … that is more flexible and robust
  - … and works in both remote and local access modes



# Who is using REST for cloud storage?

- Tons!
  - Amazon S3, Eucalyptus, Rackspace, Mezeo, Nirvanix, ...etc
- Cloud storage abstractions libraries ease transition
  - jClouds for java/clojure
  - Simple Cloud for PHP


# Cloud storage DYI with Infinispan

- No need to rely on 3rd party
- You can build your own based on Infinispan
  - Infinispan is an in-memory data grid plattform
  - Consistent hash based data distribution
  - Eviction and cache persistence
  - GUI management and monitoring provided by JOPR



# Cloud storage DYI with Infinispan

- Infinispan is remotely accessible via:
  - Hot Rod - a custom binary protocol
  - Memcached text protocol
  - WebSocket for javascript clients
  - RESTful interface (of course!)
    - WAR module deployable in servlet container



# Infinispan RESTful interface

- PUT /{cacheName}/{cacheKey}
  - Insert or update payload in cache's key
- POST /{cacheName}/{cacheKey}
  - Same as PUT but if key exists, returns HTTP conflict
- Headers for PUT/POST
  - Content-Type mandatory – use media/mime-types
  - Optional: timeToLiveSeconds, maxIdleTimeSeconds, performAsync



# Infinispan RESTful interface

- GET /{cacheName}/{cacheKey}
  - Return value of cached key as body in response
  - ETag with last modified time returned to enable caching
- DELETE /{cacheName}/{cacheKey}
  - Remove given key from cache
- DELETE /{cacheName}
  - Remove ALL the entries from given cache



# Infinispan REST Clients

- Any HTTP client will do, example:

```
#
# Sample python code using the standard http lib only
#

import httplib


#putting data in
conn = httplib.HTTPConnection("localhost:8080")
data = "SOME DATA HERE !" #could be string, or a file...
conn.request("POST", "/infinispan/rest/Bucket/0", data,
   {"Content-Type": "text/plain"})
response = conn.getresponse()
print response.status

#getting data out
import httplib
conn = httplib.HTTPConnection("localhost:8080")
conn.request("GET", "/infinispan/rest/Bucket/0")
response = conn.getresponse()
print response.status
print response.read()
```



# Applying RESTful principles to other APIs

- Any Resource Oriented Architecture (ROA)
- Where Resource is the “epicentre” of the system
  - i.e. A library with books and members as resources
- A resource has...
  - Unique ID (and hence an URI)
    - i.e. http://my.library.com/book/1234



# Applying RESTful principles to other APIs

- A resource can...
  - Be related/linked via hypermedia
  - Have different state and representations
    - i.e. Borrowed/Available, JSON/HTML/XML...etc
- Systems with single entry point and rest discovered
  - i.e. Library as entry point and books/members linked



# Applying RESTful principles to other APIs

- Business processes represented HTTP methods
  - i.e. Borrowing a book represented as:
    - PUT http://my.library.com/member/5678/book/1234
- Use self-descriptive messages
  - i.e. application/my.library.com.Book+json



# If REST told you to jump off a bridge, would you?

- Complex queries with many parameters are tricky
  - i.e. "*List the female Spanish members aged between 20 and 30 that borrowed books between 1980 and 1990, and never returned them*"
- Do not make resources out of such queries
- RPC-like query operation is much better suited here



# Summary

- REST suits cloud management APIs
  - But remember limitations with history and queries
- REST and cloud storage APIs are made for each other
- Cloud storage with Infinispan REST module is simple
- RESTful principles are applicable to other APIs
  - Don't convert queries into resources, use RPC



# Questions?

- Project site: http://www.infinispan.org
- Blog: http://blog.infinispan.org
- Twitter:
  - http://twitter.com/infinispan
    - #infinispan
  - http://twitter.com/galderz



# FOLLOW US ON TWITTER

www.twitter.com/redhatsummit

# TWEET ABOUT IT

#summitjbw

# READ THE BLOG

http://summitblog.redhat.com/