REST Overview

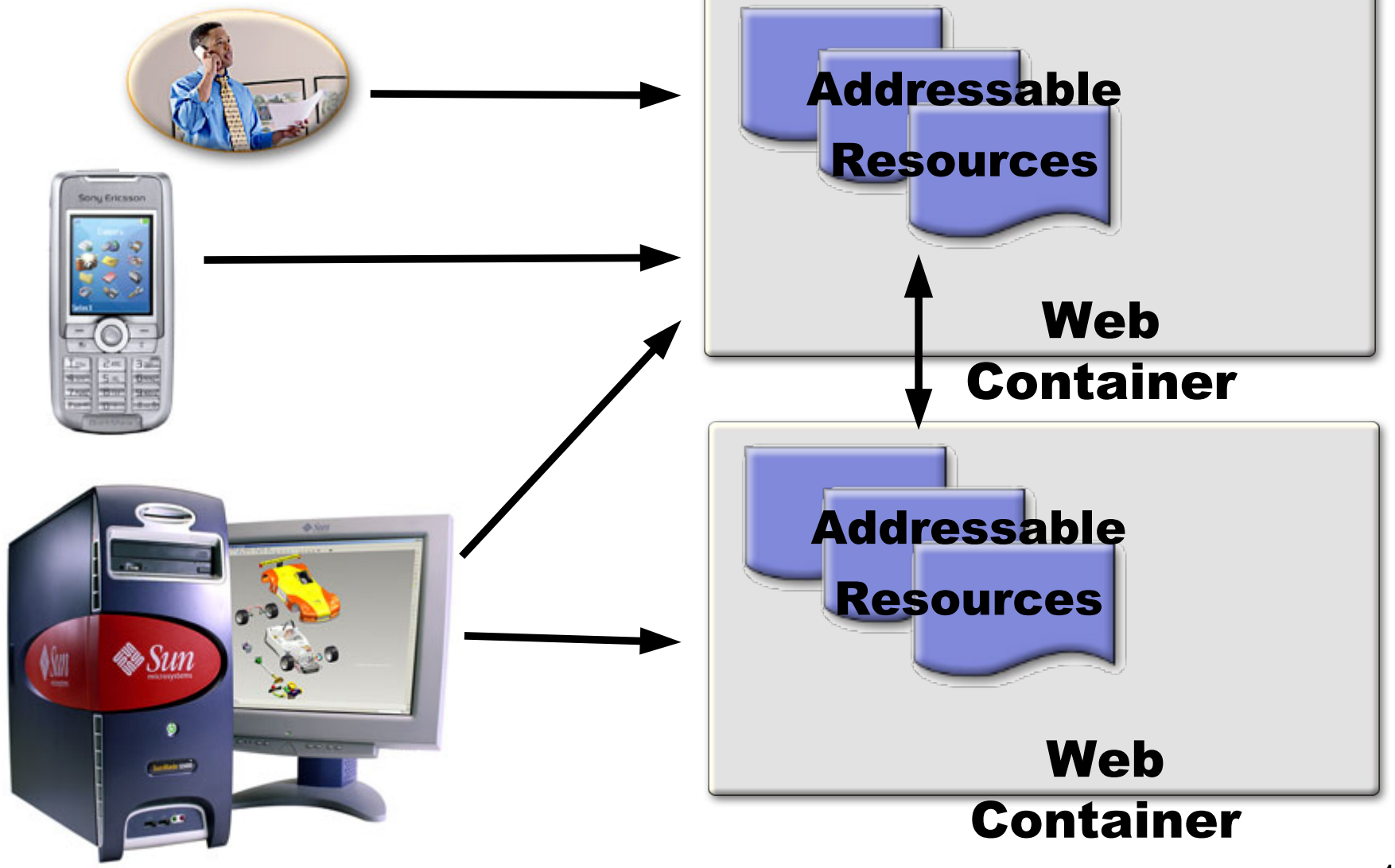
Sang Shin
JPassion.com
"Learn with Passion!"

Agenda

- How we use Web today?
- What is REST?
  - HTTP Methods
  - Common Patterns
- Why REST?
- REST vs. SOAP

How We Use Web Today?

The Web - today



Everything is accessed through URLS

http://www.sun.com/servers/blades/t6300/index.jsp

What is REST?

REST

- REpresentational State Transfer
- Name coined by Roy Fielding in his Ph.D thesis*
- Architectural Style of the Web

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

Characteristics of REST

- RESTful services have a uniform interface through well-known HTTP methods
  - GET, POST, PUT, and DELETE
- REST-based architectures are built from resources (pieces of information) that are uniquely identified by URIs
- RESTful services are stateless
  - Each request from client to server must contain all the information necessary to understand the request

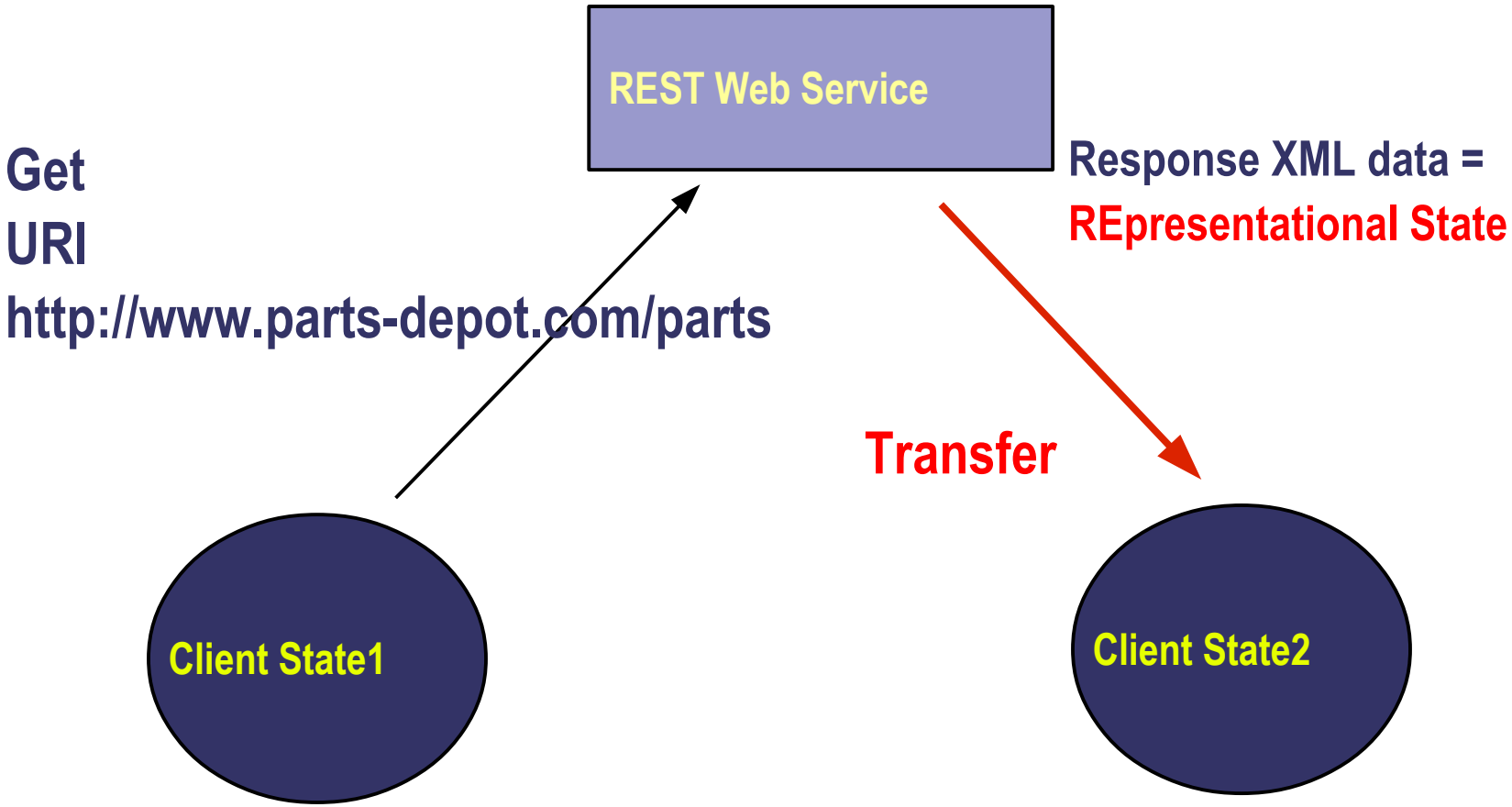
Characteristics of REST (Continued)

- In REST system, resources are manipulated through the exchange of “representations” of the resources
  - For example, a purchase order resource is represented by an XML or JSON document.
  - In a RESTful purchasing system, each purchase order is made through a combination of HTTP POST method with XML document, which represents the order, sent to a unique URI

Characteristics of REST (Continued)

- In REST system, communication occurs primarily through the transfer of representations of resources
  - State is maintained within a resource representation
  - REpresentational State Transfer

REpresentational State Transfer



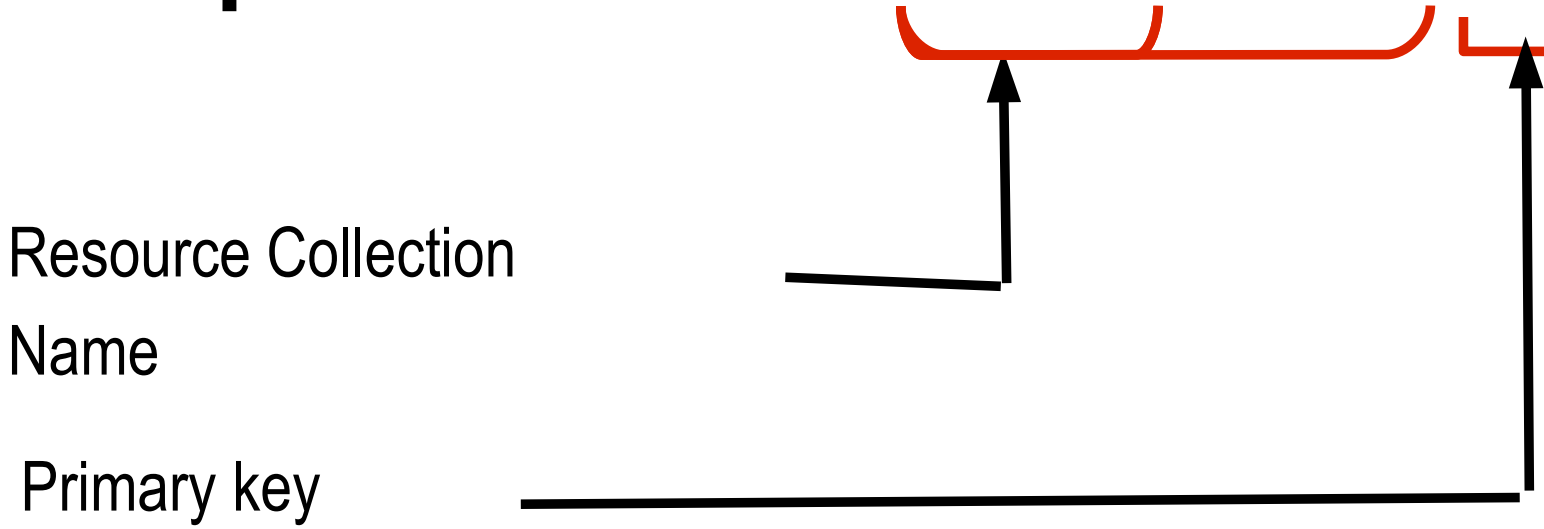
Everything (Every Resource) has an ID

- ID is a URI

  http://example.com/widgets/foo
  http://example.com/customers/bar
  http://example.com/customers/bar/orders/2
  http://example.com/orders/101230/customer

Example URI

http://www.sun.com/servers/blades/t6300



- Define how the URI maps to your resources

Multiple Representations (Formats)

- Offer data in a variety of formats
  - XML
  - JSON
  - (X)HTML
- Content negotiation can be done in two forms
  - Accept header

```
GET /foo
Accept: application/json
```

  - URI-based

```
GET /foo.json
```

What Makes Up a REST Request?

- Resources (nouns)
  - Identified by a URI, For example:
    - http://www.parts-depot.com/parts
- Methods (verbs) to manipulate the nouns
  - Small fixed set: GET, POST, PUT, and DELETE
- Representation is how you view the State
  - data and state transferred between client and server
  - XML, JSON...
- Use verbs to exchange application state and representation

HTTP Request/Response As REST

Request

```
GET /music/artists/beatles/recordings HTTP/1.1
Host: media.example.com
Accept: application/xml
```

Method

Resource

Response

```
HTTP/1.1 200 OK
Date: Tue, 08 May 2007 16:41:58 GMT
Server: Apache/1.3.6
Content-Type: application/xml; charset=UTF-8

<?xml version="1.0"?>
<recordings xmlns="…">
  <recording>…</recording>
  …
</recordings>
```

State transfer

Representation

HTTP Methods

What is REST?

CRUD Operations are Performed through HTTP method + URI

CRUD Operations	4 main HTTP methods	
	Verb	**Noun**
Create (Single)	POST	Collection URI
Read (Multiple)	GET	Collection URI
Read (Single)	GET	Entry URI
Update (Single)	PUT	Entry URI
Delete (Single)	DELETE	Entry URI

HTTP Methods: GET

- GET to retrieve information
- Retrieves a given URI
- Idempotent, should not initiate a state change
- Cacheable

HTTP Methods: POST

- POST to add new information
- Add the entity as a subordinate/append to the POSTed resource
  - > *POST /music/beatles*

Adds the music specified in the POSTDATA to the list of music

HTTP Methods: PUT

- PUT to update information
- Full entity create/replace used when you know the “id”
  - > *PUT /songs/123-456789012*

Replaces the representation of the song with an updated version.

HTTP Methods: DELETE

- Remove (logical) an entity
  - > *DELETE /songs/heyjude*

Deletes the song 'heyjude"
from the system

Common Patterns

Common Patterns: Container, Item
Server in control of URI path space

- List container contents: GET /items
- Add item to container: POST /items
  - with item in request
  - URI of item returned in HTTP response header
  - e.g. Location: http://host/items/itemid
- Read item: GET /items/itemid
- Update item: PUT /items/itemid
  - with updated item in request
- Remove item: DELETE /items/itemid
- Good example: Atom Publishing Protocol

Common Patterns: Map, Key, Value
Client in control of URI path space

- List key-value pairs: **`GET /map`**
- Put new value to map: **`PUT /map/{key}`**
  - with entry in request
  - e.g. **`PUT /map/dir/contents.xml`**

- Read value: **`GET /map/{key}`**
- Update value: **`PUT /map/{key}`**
  - with updated value in request
- Remove value: **`DELETE /map/{key}`**

- Good example: Amazon S3

Why REST?

Why REST ?

- REST base services are easy to work with
  - Clients do not need to use specialized API
  - Clients just use standard HTTP
  - You can use browser for experimentation and testing
- Uniformity
  - URI is a uniform way to identify resources
  - HTTP methods provide uniform interface to manipulate resources
- Scripting language friendly
  - Easy to consume and develop

REST vs. SOAP

REST vs SOAP

- SOAP-based web service
  - Few URIs (nouns), many custom methods (verbs)
    - `musicPort.getRecordings("beatles")`
  - Uses HTTP as transport for SOAP messages
- RESTful web service
  - Many resources (nouns), few fixed methods(verbs)
    - `GET /music/artists/beatles/recordings`
  - HTTP is the protocol

SOAP Service and REST Resource

- SOAP based web services is about services
  - Stock quote service

  quoteService.**purchase**(“sunw”, 2000, 6.0f);

- REST is Resource-Oriented Architecture
  - Stock quote resource
  - Resources are manipulated by exchanging representations
  - Eg. **purchasing** stock
    - Manipulate my portfolio resource
    - Handle a POST in a stock resource that I own
    - **`POST /mystocks/sunw`**

Advantages of SOAP over REST

- Transport independence
  - You can use SOAP over any kind of transport (HTTP/S, JMS, SMTP) while REST works only over HTTP/S
- Well-defined standards
  - SOAP has well-defined standards in the area of security, transaction, reliability while REST lacks the standards in these areas
  - SOAP better suits with stateful operations
- Well-defined service contact via WSDL
  - REST now has WADL

Advantages of REST over SOAP

- Simplicity
- REST works with different representations (XML, JSON, XHTML, etc) while SOAP works only with XML (unless you use different encoding style)

Learn with Passion!
JPassion.com