Web Services

- A system designed to support interoperability of systems connected over a network
  - Service oriented architecture (SOA)
  - A standardized way of integrating web-based applications using open standards operating over the Internet
- Two common approaches used in practice:
  - SOAP (Simple Object Access Protocol) based services
    - Uses WSDL (Web Services Description Language)
    - XML based
  - REST (Representational State Transfer)
    - Use Web standards
    - Exchange of data using either XML or JSON
    - Simpler compared to SOAP, WSDL etc.

Representational State Transfer (REST)

- A style of software architecture for distributed hypermedia systems such as the World Wide Web.
- Introduced in the doctoral dissertation of Roy Fielding
  - One of the principal authors of the HTTP specification.
- A collection of network architecture principles which outline how resources are defined and addressed

Representational State Transfer (REST)

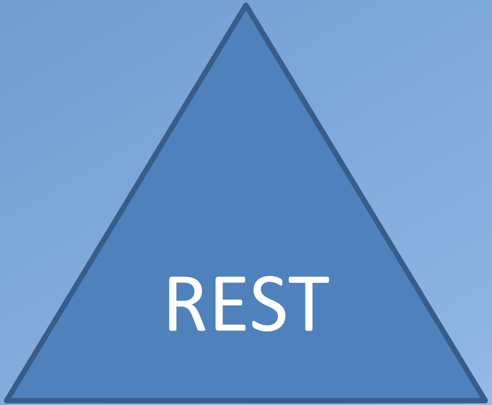
- Four basic design principles:
  - Use HTTP methods explicitly
  - Be stateless
  - Expose directory structure-like URIs
  - Transfer using XML, JavaScript Object Notation (JSON), or both

REST and HTTP

- The motivation for REST was to capture the characteristics of the Web that made the Web successful
  - URI (Uniform Resource Indicator) Addressable resources
  - HTTP Protocol
  - Make a Request – Receive Response – Display Response
- Exploits the use of the HTTP protocol beyond HTTP POST and HTTP GET
  - HTTP PUT, HTTP DELETE
  - Preserve Idempotence

REST Concepts

Nouns (Resources)
unconstrained
i.e., http://www.conFusion.food/dishes/123



Verbs
constrained
i.e., GET, PUT, POST, DELETE

Representations
constrained
i.e., XML, JSON

Resources

- The key abstraction of information in REST is a resource.
- A resource is a conceptual mapping to a set of entities
  - Any information that can be named can be a resource: a document or image, a temporal service (e.g. "today's weather in Hong Kong"), a collection of other resources, a non-virtual object (e.g. a person), and so on
- Represented with a global identifier (URI in HTTP)
  - http://www.conFusion.food/dishes/123

Naming Resources

- REST uses URI to identify resources
  - http://www.conFusion.food/dishes/
  - http://www.conFusion.food/dishes/123
  - http://www.conFusion.food/promotions/
  - http://www.conFusion.food/leadership/
  - http://www.conFusion.food/leadership/456
- As you traverse the path from more generic to more specific, you are navigating the data
- Directory structure to identify resources

Verbs

- Represent the actions to be performed on resources
  - Corresponding to the CRUD operations
- HTTP GET ←→ READ
- HTTP POST ←→ CREATE
- HTTP PUT ←→ UPDATE
- HTTP DELETE ←→ DELETE

HTTP GET

- Used by clients to request for information
- Issuing a GET request transfers the data from the server to the client in some representation (XML, JSON)
  - GET http://www.conFusion.food/dishes/
    - Retrieve all dishes
  - GET http://www.conFusion.food/dishes/452
    - Retrieve information about the specific dish

HTTP PUT, HTTP POST, HTTP DELETE

- HTTP POST creates a resource
  - POST http://www.conFusion.food/feedback/
    - Content: {first name, last name, email, comment etc.}
    - Creates a new feedback with given properties
- HTTP PUT updates a resource
  - PUT http://www.conFusion.food/dishes/123
    - Content: {name, image, description, comments …}
    - Updates the information about the dish, e.g., comments
- HTTP DELETE removes the resource identified by the URI
  - DELETE http://www.conFusion.food/dishes/456
    - Delete the specified dish

Representations

- How data is represented or returned to the client for presentation
- Two main formats:
  - JavaScript Object Notation (JSON)
  - XML
- It is common to have multiple representations of the same data
  - Client can request the data in a specific format if supported

Stateless Server

- Server side should not track the client state:
  - Every request is a new request from the client
- Client side should track its own state:
  - E.g., using cookies, client side database
  - Every request must include sufficient information for server to serve up the requested information
  - Client-side MVC setup

REST?

- The REST is not history!