**Representational State Transfer**

Web services

* A way of designing systems to support interoperability among systems that are connected over a network (e.g., the internet)
* 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
* 2 common approaches used for supporting web services
  + 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 either using XML or JSON
    - Simpler compared to SOAP, WSDL, etc.
    - REST has found a wider deployment in the web services world

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 can be made available on servers. These resources can be accessed by clients
* 4 basic design principles
  + Use HTTP methods explicitly
  + Be stateless
  + Expose directory structure-like URLs
  + Transfer using XML, JSON, or both
* REST and HTTP
  + The motivation for REST was to capture the characteristics of the Web that made the Web successful
    - The use of URI(Uniform Resource Indicator) Addressable resources
    - HTTP protocol
    - Request response cycle: 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 – can be repeated multiple times but still have the same result
* REST concepts (unconstrained)
  + Nouns
    - Resources ex: http://www.facebook.com/giodivino/123
  + Verbs (constrained)
    - GET
    - POST
    - DELETE
  + Representations (constrained)
    - XML, JSON
* Resources
  + The key abstraction of information in REST is a resource
    - The information is abstracted in the form of a resource and a resource is specified by using a URL
    - Any information that can be encapsulated and be made available can be made into a resource
  + A resource is a conceptual mapping to a set of entities
    - Any information that can be named can be a resource
  + Represented w a global identifier (URI in HTTP)
  + Naming resources
    - REST uses URI to identify resources
      * <http://www.conFusion.food/dishes/>
      * <http://www.conFusion.food/dishes/123>
      * <http://www.conFusion.food/promotions>
    - As you traverse the path from more generic to more specific, you are navigating the data
    - This is the directory structure that we use
* Verbs
  + Represent actions to be performed on resources
    - Corresponding to the CRUD operations
  + HTTP GET -> READ
    - You want to perform a read operation on the source
    - Wants to obtain a representation of the source
    - 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)
  + HTTP POST -> CREATE
    - You want to create a new resource and specify the details of the resource in the representation that is used for specifying the resource and send the information over to the server side and so the server will create the resource on your behalf
    - Creates a new resource on the server side
  + HTTP PUT -> UPDATE
    - Modification of resources
    - Used to update a resource
  + HTTP DELETE -> DELETE
    - Deletion of resources
* Representations
  + How data is represented or returned to the client for presentation
  + 2 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 is a new request from the client
  + Client side should track its own state
    - E.g., using cookies and client - side database
    - Every request must include sufficient information for server to serve up the requested information
    - Client-side MVC setup