Web Services

Daniel Zappala

CS 360 Internet Programming
Brigham Young University

Web Services

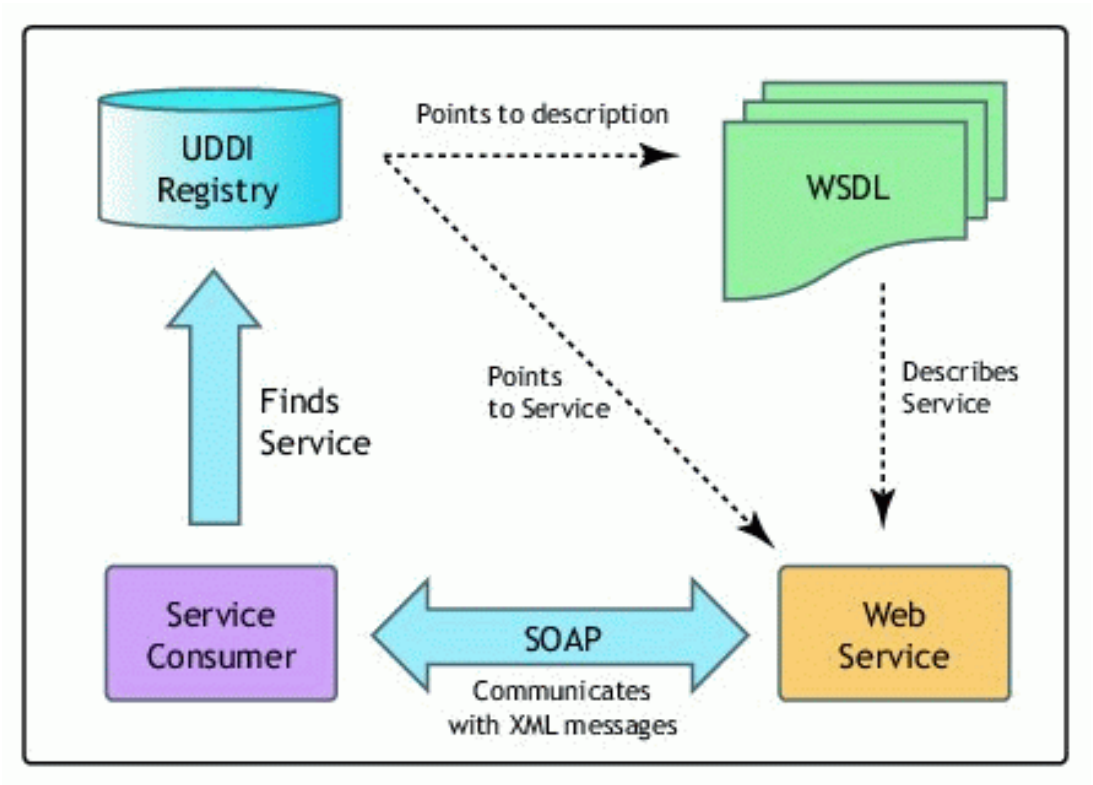
- purpose
  - a programmatic way to interact with a web application
  - allows third-party software to interact with a web application
  - mobile apps, Twitter bots, etc.
- REST/JSON
  - lightweight, simple, cacheable
  - built on HTTP
- SOAP/WSDL/UDDI/XML
  - heavyweight, complex
  - W3C standards, industry support

SOAP

Service Oriented Architecture

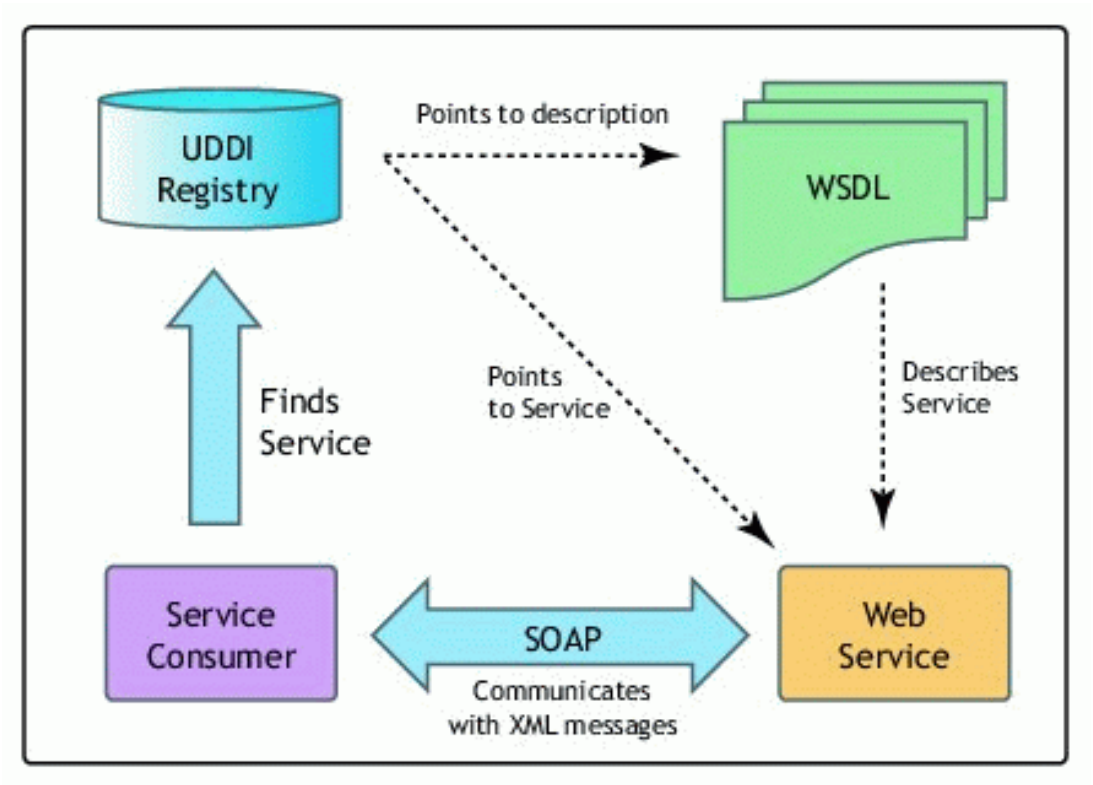
- *loose coupling among interacting software agents*
  - agents are generally programs, not users
  - separate data from computing and viewing
- example
  - a company needs to ship some packages overseas, so it uses a program to look up package delivery services, compare prices, purchase the best deal, and schedule pickup
- requires
  - service discovery
  - interfaces
  - standardized and extensible protocols

Web Services Architecture



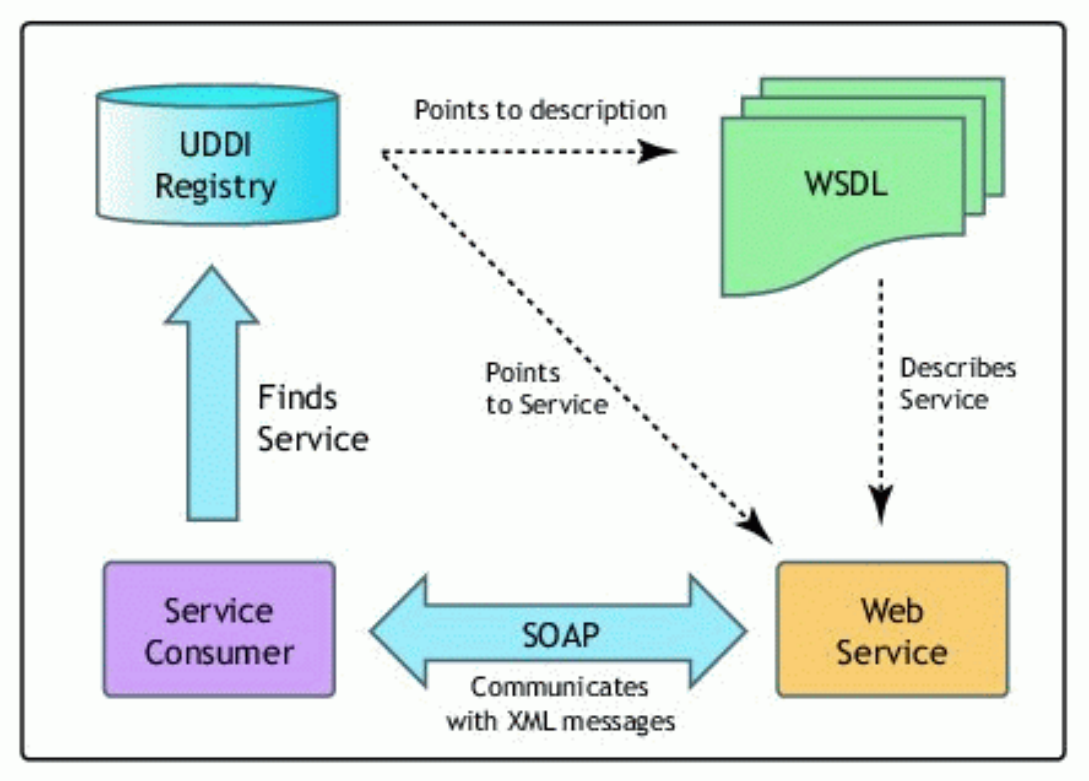
- UDDI: Universal Discovery, Description and Integration
  - platform-independent, XML-based registry listing available web services
  - a place where service providers can advertise available services and do business with partners

Web Services Architecture



- WSDL: Web Services Description Language
  - XML format for describing web services
  - standardized by W3C: Web Services Description Working Group
  - example: see Section 2.1 of the WSDL Version 2.0 Primer

Web Services Architecture



- SOAP: Simple Object Access Protocol
  - protocol for obtaining services using XML messages
  - description of service must be in WSDL

SOAP Request

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    >
  <soap:Body>
    <getProductDetails xmlns="http://warehouse.example.com/ws">
      <productID>827635</productID>
    </getProductDetails>
  </soap:Body>
</soap:Envelope>
```

SOAP Response

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <getProductDetailsResponse xmlns="http://warehouse.example.com/ws">
      <getProductDetailsResult>
        <productName>Toptimate 3-Piece Set</productName>
        <productID>827635</productID>
        <description>3-Piece luggage set. Black Polyester.</description>
        <price>96.50</price>
        <inStock>true</inStock>
      </getProductDetailsResult>
    </getProductDetailsResponse>
  </soap:Body>
</soap:Envelope>
```

REST

Representational State Transfer (REST)

- web services using the existing web architecture
  - observation: everything we need to do to invoke a web service is already supported in HTTP
  - simply need to add XML or JSON formats for results
- based on the concept of a resource, identified by a URI
- use standard HTTP methods to access a resource
  - GET: obtain a representation of a resource
  - DELETE: remove a representation of a resource
  - POST: update or create a representation of a resource
  - PUT: create a representation of a resource
- compare to SOAP, where each application defines its own custom methods

REST Example: FamilySearch

```
https://familysearch.org/platform/tree/ancestry
```

- specify
  - Authorization header with OAuth token
  - Accept header with desired format (JSON, XML)
  - starting person
  - whether to include ancestry of spouse
  - number of generations
- returns
  - a set of persons in the tree
- ▸ Example

Available APIs

- GitHub
- Twitter
- goodreads
- ToodleDo
- Google