

# Web Services

---

hussein suleman  
uct csc3002f 2007

## Definitions

---

- Web
  - = World Wide Web
  - = client-server hypermedia system layered over Internet.
- Web service
  - = Web-based Service (or service)
  - = service provided through the medium of the Web, beyond hypermedia.
- Web Service
  - = machine-to-machine communication based on interoperability standards defined by W3C/WSI.

## SOA and Web Services

---

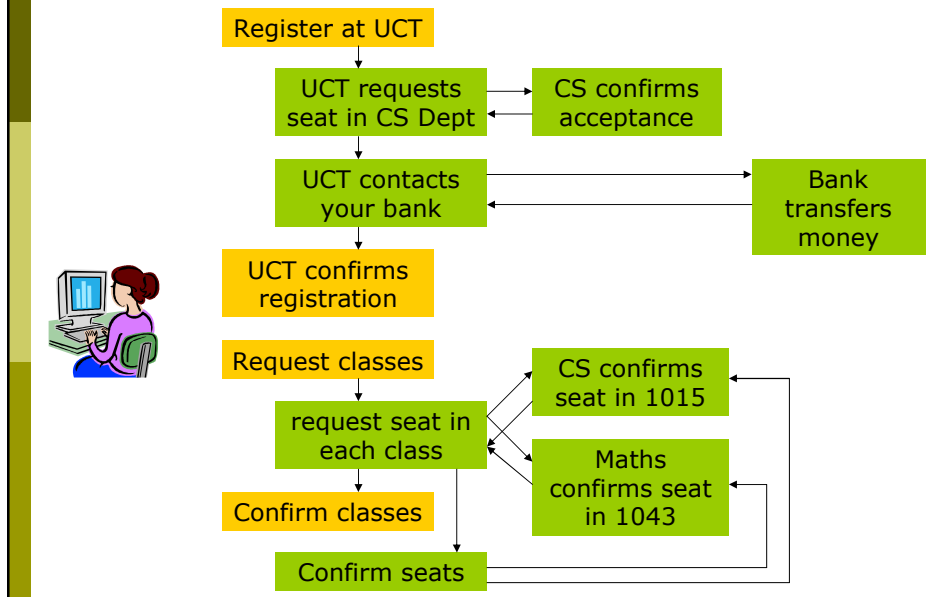
- Service Oriented Architectures (SOA) are a methodology in software development where software is cast as a set of services with well-defined external interfaces.
  - Benefits include:
    - modularity
    - reuse
- Web Services are a specific set of SOA standards for the description of machine-to-machine communication (especially over the Web):
  - SOAP
  - WSDL
  - UDDI

## REST as an alternative

---

- Representational State Transfer is an alternative philosophy to SOA.
- REST suggests that a resource should ideally be read with only a GET HTTP request, without the overhead of Web Services!
  - along with other things ...

## Use case: Registration



## Mapping to Core Web Services

- SOAP
  - Transport protocol for message passing.
- WSDL
  - Service interface description for BANK, UCT, Dept1, Dept2, CS.
- UDDI
  - Central registry of all services e.g., CS's "did we accept you" service.
  
- Web-based services for user interaction!

## SOAP

---

- Describes an XML format for messages to be exchanged among actors.
  - One-way information transfer.
  - Stateless.
  - Independent of lower-layer transport.
  - Specifies only syntax, not semantics.
  
- Used to be "Simple Object Access Protocol".

## SOAP Example

---

```
<Envelope>
  <header>
    <origin>CS</origin>
  </header>
  <body>
    <requestSeat>
      <class>CSC115</class>
      <studentID>abc123</studentID>
    </requestSeat>
  </body>
</Envelope>
```

## SOAP Actors and Roles

---

- Producer
  - Creates SOAP message.
- Consumer
  - Uses/interprets/understands SOAP message.
- Intermediary
  - Modifies message in transmission path between producer and consumer.
  
- Role attributes can specify if message block must be understood by next or final node.

## SOAP headers and bodies

---

- Headers may contain control information that may be modified by intermediaries.
  - Can have multiple headers.
  - Headers can be modified, removed and inserted by intermediaries.
- Body contains application payload.
  - Only one body per message.
  
- Many simple use cases do not “need” headers.
- Both are defined using XML Schema, as specified in WSDL.

## WSDL

---

- Web Services Description Language defines interactions among WS nodes in terms of:
  - corresponding message pairs or sequence of messages.
  - message types.
  - protocol binding (HTTP or SOAP).
  - network address of service.
- Using a WSDL definition, a client can determine how to use a Web Service (syntactically).

## UDDI

---

- The Universal Description, Discovery and Integration of Web Services (UDDI) is a central/replicated clearinghouse for Web Services.
- UDDI is not machine-readable – needs human intervention to select services.
- Alternatives now include P2P discovery, automatic indices, etc.

# UDDI Example 1/4

The screenshot shows the IBM UDDI Business Registry Version 2 search page. The page has a blue header with the IBM logo and a search bar. Below the header is a navigation menu with links for Home, Products & services, Support & downloads, and My account. The main content area is titled "UDDI Business Registry Version 2" and "UDDI Find". It includes a "Simple Search" section with a search form and an "Advanced Search" section with links for "Find a Business", "Find a Service", and "Find a Technical Model".

**UDDI Business Registry**  
Find

**UDDI Business Registry Version 2**  
Universal Description, Discovery, and Integration

**UDDI Find**

**Simple Search**  
Simple Search allows you to search for a Business, Service, or Technical Model by NAME and / or CATEGORY.  
You may use the '%' symbol as a wildcard that matches any character.

Search For a

Starting with

Category

Values [Add Locator](#)

**Advanced Search**  
Advanced Search allows you to perform complex searches using multiple criteria as well as override default search behaviors.

[Find a Business](#) [Find a Service](#) [Find a Technical Model](#)

[Help](#) [Terms of Use](#) [Support](#)

[About IBM](#) [Privacy](#) [Legal](#) [Contact](#)

# UDDI Example 2/4

The screenshot shows the IBM UDDI Business Registry Version 2 search results page. The page displays the search results for the query "amazon". It includes a table of services with columns for Service Name, Description, and Owning Business. The results show three matching services: Amazon Web Services 2.0, Amazon.com Web Services, and AmazonBookPrice.

**UDDI Business Registry**  
Find

**UDDI Business Registry Version 2**  
Universal Description, Discovery, and Integration

**Find Service Results**


Your query returned a total of 3 matching service(s). Press the **New Search** button to search again.

Service Name	Description	Owning Business
<a href="#">Amazon Web Services 2.0</a>	At Amazon, we want to see Web services work. We believe they are important to the future of the Internet. To help stimulate Web service innovation, we now offer software developers the opportunity to integrate Amazon.com and Amazon.co.uk features and cont	<a href="#">Amazon Web Services for Testing</a>
<a href="#">Amazon.com Web Services</a>	Set of web services that allow developers to create applications that consume Amazon.com core features. When tied to Amazon.com Associate program, developers can earn a percentage of each transaction that Amazon.com fulfills. Developers must have a token	<a href="#">Amazon.com</a>
<a href="#">AmazonBookPrice</a>	Amazon Book Price Service	<a href="#">RAIC Test</a>

# UDDI Example 3/4

**UDDI Business Registry**  
Find

**Related Links:**  
[Web Services and UDDI](#)  
[IBM UDDI Business Test Registry](#)

**IBM partners:**  


## UDDI Business Registry Version 2

Universal Description, Discovery, and Integration

### Service Details

The details of the selected service are shown below. Please use your browser's **Back** button to return to the previous page OR Press the **New Search** button to search again.

Service Information	
<b>Key</b>	BA6D9D56-EA3F-4263-A95A-EEB17E5910DB
<b>Owning Business</b>	<b>Owner Key</b>
Amazon.com	18B7FDE2-D15C-437C-8877-EBEC8216D0F5

Service Name(s)	
<b>Name</b>	<b>Language</b>
Amazon.com Web Services	en

Service Description(s)	
<b>Description</b>	<b>Language</b>
Set of web services that allow developers to create applications that consume Amazon.com core features. When tied to Amazon.com Associate program, developers can earn a percentage of each transaction that Amazon.com fulfills. Developers must have a token	en

Access Point(s)			
<b>Protocol</b>	<b>Address</b>	<b>Description</b>	<b>Actions</b>
http	<a href="http://soap.amazon.com/schemas/AmazonWebServices.wsdl">http://soap.amazon.com/schemas/AmazonWebServices.wsdl</a>	The WSDL file that allows developers to	<a href="#">Details</a>

# UDDI Example 4/4 (WSDL)

```

<!--
  interfaces are subject to
-->
<definition name="AmazonSearch" targetNamespace="urn:PI/DevCentral/SoapService">
<!--
  Data structures used in Amazon.com's Web Service calls and responses
-->
<type>
  <xsd:schema targetNamespace="urn:PI/DevCentral/SoapService">
    <xsd:complexType name="ProductInfo">
      <xsd:attribute
        <xsd:element name="Details" type="typens:DetailsArray"/>
      </xsd:attribute>
    </xsd:complexType>
  <xsd:complexType name="Details">
    <xsd:attribute
      <xsd:element name="Url" type="xsd:string"/>
      <xsd:element name="Asin" type="xsd:string"/>
      <xsd:element name="ProductName" type="xsd:string"/>
      <xsd:element name="Catalog" type="xsd:string"/>
      <xsd:element name="KeyPhrases" type="typens:KeyPhraseArray"/>
      <xsd:element name="Artists" type="typens:ArtistArray"/>
      <xsd:element name="Authors" type="typens:AuthorArray"/>
    
```

## Example: Google Web Search API

- Until 2006, Google offered a free Web Service interface.
  - Up to 1000 requests per day per user.
  - Defined using WSDL, using HTTP-SOAP.
- Every user must first request a unique key to track user activity, and provide this key with every request.
- Requests can be for search results, cached data or spelling checks.

## Google WSDL 1/2

```
<xsd:complexType name="GoogleSearchResult">
  <xsd:all>
    <xsd:element name="documentFiltering" type="xsd:boolean"/>
    <xsd:element name="searchComments" type="xsd:string"/>
    <xsd:element name="estimatedTotalResultsCount" type="xsd:int"/>
    <xsd:element name="estimateIsExact" type="xsd:boolean"/>
    <xsd:element name="resultElements" type="typens:ResultEl"/>
    <xsd:element name="searchQuery" type="xsd:string"/>
    <xsd:element name="startIndex" type="xsd:int"/>
    <xsd:element name="endIndex" type="xsd:int"/>
    <xsd:element name="searchTips" type="xsd:string"/>
    <xsd:element name="directoryCategories" type="typens:Director"/>
    <xsd:element name="searchTime" type="xsd:double"/>
  </xsd:all>
</xsd:complexType>

<xsd:complexType name="ResultElement">
  <xsd:all>
    <xsd:element name="summary" type="xsd:string"/>
    <xsd:element name="URL" type="xsd:string"/>
    <xsd:element name="snippet" type="xsd:string"/>
    <xsd:element name="title" type="xsd:string"/>
    <xsd:element name="cachedSize" type="xsd:string"/>
    <xsd:element name="relatedInformationPresent" type="xsd:boolean"/>
    <xsd:element name="hostName" type="xsd:string"/>
    <xsd:element name="directoryCategory" type="typens:DirectoryCategory"/>
    <xsd:element name="directoryTitle" type="xsd:string"/>
  </xsd:all>
</xsd:complexType>
```

## Google WSDL 2/2

---

```
<message name="doGoogleSearch">
  <part name="key" type="xsd:string"/>
  <part name="q" type="xsd:string"/>
  <part name="start" type="xsd:int"/>
  <part name="maxResults" type="xsd:int"/>
  <part name="filter" type="xsd:boolean"/>
  <part name="restrict" type="xsd:string"/>
  <part name="safeSearch" type="xsd:boolean"/>
  <part name="lr" type="xsd:string"/>
  <part name="ie" type="xsd:string"/>
  <part name="oe" type="xsd:string"/>
</message>

<operation name="doGoogleSearch">
  <input message="typens:doGoogleSearch"/>
  <output message="typens:doGoogleSearchResponse"/>
</operation>
```

## Google SOAP Request

---

```
<SOAP-ENV:Envelope xmlns:SOAP-
  ENV="http://schemas.xmlsoap.org/soap/envelope/" xm
  lns:xsi="http://www.w3.org/1999/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org
  /1999/XMLSchema">
  <SOAP-ENV:Body>
    <ns1:doGoogleSearch xmlns:ns1="urn:GoogleSearch"
      SOAP-
      ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <key
        xsi:type="xsd:string">00000000000000000000000000000000</key>
      <q xsi:type="xsd:string">uct computer science</q>
      <start xsi:type="xsd:int">0</start>
      <maxResults xsi:type="xsd:int">10</maxResults>
      <filter xsi:type="xsd:boolean">>false</filter>
      <restrict xsi:type="xsd:string"></restrict>
      <safeSearch xsi:type="xsd:boolean">>false</safeSearch>
      <lr xsi:type="xsd:string"></lr>
      <ie xsi:type="xsd:string">latin1</ie>
      <oe xsi:type="xsd:string">latin1</oe>
    </ns1:doGoogleSearch>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

## Google SOAP Response 1/2

```
<SOAP-ENV:Envelope>
  <SOAP-ENV:Body>
    <ns1:doGoogleSearchResponse>
      <return xsi:type="ns1:GoogleSearchResult">
        <documentFiltering xsi:type="xsd:boolean">>false</documentFiltering>
        <estimatedTotalResultsCount
          xsi:type="xsd:int">3</estimatedTotalResultsC
ount>
        <directoryCategories
          xmlns:ns2="http://schemas.xmlsoap.org/soap/encoding
/" xsi:type="ns2:Array"
          ns2:arrayType="ns1:DirectoryCategory[0]"></directoryCate
gories>
          <searchTime xsi:type="xsd:double">0.194871</searchTime>
          <resultElements
            xmlns:ns3="http://schemas.xmlsoap.org/soap/encoding/" xs
i:type="ns3:Array" ns3:arrayType="ns1:ResultElement[3]">
            <item xsi:type="ns1:ResultElement">
              <cachedSize xsi:type="xsd:string">12k</cachedSize>
              <hostName xsi:type="xsd:string"></hostName>
              <snippet xsi:type="xsd:string"> some stuff ... </snippet>
              <directoryCategory xsi:type="ns1:DirectoryCategory">
                <specialEncoding xsi:type="xsd:string"></specialEncoding>
                <fullViewableName xsi:type="xsd:string"></fullViewableName>
              </directoryCategory>
            </item>
          </resultElements>
        </return>
      </ns1:doGoogleSearchResponse>
    </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

## Google SOAP Response 2/2

```
<relatedInformationPresent xsi:type="xsd:boolean">true</relatedInfor
mationPresent>
  <directoryTitle xsi:type="xsd:string"></directoryTitle>
  <summary xsi:type="xsd:string"></summary>
  <URL
    xsi:type="xsd:string">http://hci.stanford.edu/cs147/examples/sh
rdlu/</URL>
  <title xsi:type="xsd:string">&lt;b>SHRDLU</b></title>
  </item>
</resultElements>
<endIndex xsi:type="xsd:int">3</endIndex>
<searchTips xsi:type="xsd:string"></searchTips>
<searchComments xsi:type="xsd:string"></searchComments>
<startIndex xsi:type="xsd:int">1</startIndex>
<estimateIsExact xsi:type="xsd:boolean">true</estimateIsExact>
<searchQuery xsi:type="xsd:string">shrdlu winograd maclisp
teletype</sea
rchQuery>
</return>
</ns1:doGoogleSearchResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

## Using Google API from Perl

---

```
use SOAP::Lite;
# create soap object
my $key='00000000000000000000000000000000';
my $query="uct computer science";
my $googleSearch = SOAP::Lite -> service("file:./GoogleSearch.wsdl")
    -> proxy
    ('http://api.google.com/search/beta2',
     proxy => ['http' =>
               'http://cache.uct.ac.za:8080']);
# submit to server and get results / retry while request not accepted
my $result;
my $max_retries = 0;
my $retry_count = 0;
while (!$results->{resultElements} && $retry_count <= $max_retries)
{
    eval {
        $result = $googleSearch -> doGoogleSearch(
            $key, $query, 0, 10, "false", "", "false", "", "latin1", "latin1");
    };
    $retry_count++;
}
# print out titles and URLs
foreach my $aresult (@{$result->{'resultElements'}})
{
    print (sprintf ("%s\n    URL: %s\n", $aresult->{'title'}, $aresult->{'URL'}));
}
}
```

## Output from Google via Perl

---

```
Department of <b>Computer Science</b>, University of Cape Town
URL: http://www.cs.uct.ac.za/
Why study at the Department of <b>Computer Science</b>, University of <b>...</b>
URL: http://www.cs.uct.ac.za/newstudents/why.html
Prospective Undergrad Students for the Department of <b>Computer</b> <b>...</b>
URL: http://www.cs.uct.ac.za/newstudents/staffp.html
About Us: Department of <b>Computer Science</b>, University of Cape Town
URL: http://www.cs.uct.ac.za/about.html
Postgrad Students of the Department of <b>Computer Science</b> <b>...</b>
URL: http://www.cs.uct.ac.za/people/stud-postgrad.html
Prospective Undergrad Students for the Department of <b>Computer</b> <b>...</b>
URL: http://www.cs.uct.ac.za/people/stud-alumni.html
Admin staff of the Department of <b>Computer Science</b>, University of <b>...</b>
URL: http://www.cs.uct.ac.za/people/staff-admin.html
Department of <b>Computer Science</b>
URL: http://moodle.cs.uct.ac.za/
Using Windows CE.NET 4.1 in <b>UCT&#39;s Computer Science</b> Department
URL:
http://arc.cs.odu.edu:8080/dp9/getrecord/oai_dc/techreports.cs.uct.ac.za/oai:techre
ports.cs.uct.ac.za:65
ScienceDirect - Theoretical <b>Computer Science</b> : Scheduling UET-<b>UCT</b>
<b>...</b>
URL: http://dx.doi.org/10.1016/0304-3975(96)00035-7
```

## Beyond Google Web Search

---

- Other Google Web APIs
  - Google Maps
  - Google Earth
  - etc.
- Google RSS/Atom feeds for GMail.
  
- MSN and Yahoo Web APIs.

## Amazon Electronic Commerce

---

- A full suite of Web Services to interface with Amazon.com
  - Both SOAP and REST (URL-encoded) interfaces
  - Many more interfaces than Google. Why?
  - Fewer restrictions on usage, in production and free for many purposes!

## Amazon REST Request

---

- ❑ `http://webservices.amazon.com/onca/xml?Service=AWSECommerceService&AWSAccessKeyId=0000000000&Operation=ItemLookup&ItemId=0620056886`
  
- ❑ `baseUrl`
  - `http://webservices.amazon.com/onca/xml?Service=AWSECommerceService`
- ❑ `Key to access service`
  - `AWSAccessKeyId=000000000000`
- ❑ `Operation to perform`
  - `Operation=ItemLookup`
- ❑ `Item to display`
  - `ItemId=0620056886`

## Output from Amazon 1/2

---

```
<ItemLookupResponse>
  <OperationRequest>
    <HTTPHeaders>
      <Header Name="UserAgent" Value="Mozilla/5.0 (Windows; U; Windows
        NT 5.1; en-US; rv:1.8.0.5) Gecko/20060719 Firefox/1.5.0.5"/>
    </HTTPHeaders>
    <RequestId>0XJBJRVTVME4XENAVPQS</RequestId>
    <Arguments>
      <Argument Name="Service" Value="AWSECommerceService"/>
      <Argument Name="ItemId" Value="0620056886"/>
      <Argument Name="AWSAccessKeyId" Value="00000000000000000000"/>
      <Argument Name="Operation" Value="ItemLookup"/>
    </Arguments>
    <RequestProcessingTime>0.0196020603179932</RequestProcessingTime>
  </OperationRequest>
  <Items>
    <Request>
      <IsValid>True</IsValid>
      <ItemLookupRequest>
        <ItemId>0620056886</ItemId>
      </ItemLookupRequest>
    </Request>
```

## Output from Amazon 2/2

---

```
<Item>
<ASIN>0620056886</ASIN>
<DetailPageURL>
http://www.amazon.com/exec/obidos/redirect?tag=ws%26link_code=xm2%26camp=20
25%26creative=165953%26path=http://www.amazon.com/gp/redirect.html%253fA
SIN=0620056886%2526tag=ws%2526lcode=xm2%2526cID=2025%2526ccmID=165953%25
26location=/o/ASIN/0620056886%25253FSubscriptionId=000000000000000000
</DetailPageURL>
<ItemAttributes>
<Author>Andrew Verster</Author>
<Creator Role="Editor">Zuleikha Mayat</Creator>
<Creator Role="Photographer">Dennis Bughwan</Creator>
<Creator Role="Illustrator">Nalin Bughwan</Creator>
<Manufacturer>Domain Enterprises</Manufacturer>
<ProductGroup>Book</ProductGroup>
<Title>Indian Delights</Title>
</ItemAttributes>
</Item>
</Items>
</ItemLookupResponse>
```