Lecture 6: UDDI
Web Service Protocol Stack

Searching / Publishing Web Services

Discovery

Description

XML Messaging

Transport

Encoding messages in XML format

Transporting XML messages between client and server

Describing Web Services interface

XML-RPC, SOAP, XML

HTTP, SMTP, FTP

WSDL

UDDI
Web Services Architecture

UDDI Registry

Links to WSDL documents

Publish

Service provider business application

Search

SOAP messages

Service consumer business application
What is UDDI?

• *Universal Description, Discovery, and Integration*

• A project to speed interoperability and adoption for Web services
  – Standards-based specifications for service description and discovery
  – Shared operation of a business registry on the web

• Is a cross-industry initiative to create a registry standard for Web service description and discovery together with facility that supports publishing and discovery processes

• Is a registry that contains relatively lightweight data
Why Do We Need a Web Services Registry

• Web services are valuable because of standardized payloads and transport definitions
  – The value is creating a Web service that is used by many clients
• Can’t happen unless the services are advertised to multiple consumers
What Does UDDI Contain?

• Businesses and other service providers
• Services they expose
• Bindings (locations) of those services
• Interfaces supported by those services
What Problems Does it Solve?

**Describe Services**
- **Broader B2B**: A mid-sized manufacturer needs to create 400 online relationships with customers, each with their own set of standard and protocols.

**Discover Services**
- **Smarter Search**: A flower shop in Australia wants to be “plugged in” to every marketplace in the world, but doesn’t know how.

**Integrate Them Together**
- **Easier Aggregation**: A B2B marketplace cannot get catalog data for relevant suppliers in its industry, along with connections to shippers, insurers, etc.
Background

• Represent a technical specification for publishing and finding businesses and Web services
• UDDI 1.0 was originally announced by Microsoft, IBM and Ariba in September 2000
• In May 2001, Microsoft and IBM launched the first UDDI operator sites
• UDDI 2.0 beta was announced in June 2001
# UDDI v3 History

<table>
<thead>
<tr>
<th>VERSION</th>
<th>YEAR</th>
<th>GOALS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>2000</td>
<td>Public registry foundation</td>
</tr>
<tr>
<td>2.0</td>
<td>2003</td>
<td>Web services alignment and extensible taxonomies</td>
</tr>
<tr>
<td>3.0</td>
<td>2004</td>
<td>Flexible and secure registry interaction models</td>
</tr>
</tbody>
</table>
How did this come about?

- Naturally “evolved”
  - Ariba and IBM collaboration around B2B
  - IBM and Microsoft collaboration around XML and SOAP
  - Microsoft and Ariba collaboration around BizTalk and cXML (Corporate XML standard for data exchange between businesses)
UDDI Technical Overview

The UDDI technical architecture consists of three parts:

• UDDI data model
  – An XML Schema (a rule) for describing businesses and Web services

• UDDI API
  – A SOAP-based API for searching and publishing UDDI data

• UDDI cloud services
  – Operator sites that provide implementations of the UDDI specification (e.g. Microsoft’s UDDI site)
UDDI Implementation

UDDI Business Registry
- Programmatic descriptions of Web services
- Programmatic descriptions of businesses and the services they support
- Programming model, schema, and platform agnostic
- Uses XML, HTTP, and SOAP
- Free on the Internet

Manufacturers

Flower Shops

Marketplaces
How UDDI Works

1. Industry consortia, standard bodies, and service providers populate the registry with descriptions of different types of services.

2. Businesses populate the registry with descriptions of the services they support.

3. UBR assigns a programmaticaly unique identifier to each service and business registration.

4. Marketplaces, search engines, and business apps query the registry to discover services at other companies.

5. Business uses this data to facilitate easier integration with each other over the Web.
Registry Data

- Businesses register public information about themselves

- Standards bodies, Programmers, Businesses register information about their Service Types
White Pages

- General info about a company including
  - Business name
  - Address
  - Contact info
    - names, phone numbers, fax numbers, web sites…
  - Known Identifiers
    - list of identifiers that a business may be known by, such as Dun & Bradstreet’s D-U-N-S (Data Universal Numbering System) classifications
Yellow Pages

- General classification info about the company or service it offers

- Business categories
  - 3 standard taxonomies in V1
    - Industry: NAICS (Industry codes - US Govt.)
    - Product/Services: UN/SPSC (ECMA)
    - Location: Geographical taxonomy
  - Implemented as name-value pairs to allow any valid taxonomy identifier to be attached to the business white page
Green Pages

• New set of technical information businesses use to describe how to “do e-commerce” with them
  – Allows programmatic discovery and interaction with Web services
• Nested model
  • Business processes (providers)
  • Service descriptions
    – Owned by a business (provider)
    – A business (provider) can own multiple services
  • Binding information
    – Refers to where a service is located, how to interact with it, …
      » For SOAP-based web services, the URL
    – A service can have multiple bindings
  – Programming/platform/implementaion agnostic
  – Services can also be categorized
Service Type Registration

- Information referred to in the UDDI specifications as a tModel
- Pointer to the namespace where service type is described
  - What programmers read to understand how to use the service
  - For XML web services, the WSDL file
- Identifier for who published the service type

- Identifier for the service type registration
  - called a tModelKey
    - A unique ID generated by the UDDI registry
    - Used as a signature by Web sites that implement those services
Business Registration

- XML document
- Created by end-user company (or on their behalf)
- Can have multiple service listings
- Can have multiple taxonomy listings
Example of a Registration

businessEntity
- TB993...
- Harbour Metals
- www.harbourmetals.co.au
- “Serving Inner Sydney Harbour for ...
- contacts
- businessServices
- identifierBag
- categoryBag

Peter Smythe
- 872-6891
- 4281 King’s Blvd, Sydney, NSW
- Peter@harbourmetals.co.au

businessService
- 23T701e54683nf...
- Online catalog
- “Website where you can ...”
- BindingTemplates

keyedReference
- EE123...
- NAICS 02417

keyedReference
- DFE-2B...
- DUNS 45231

BindingTemplate
- 5E2D412E5-44EE-...
- http://www.sydney.net/harbour...
- tModelInstanceDetails

BindingTemplate
- 4453D6FC-223C-3ED0...
- http://www.rosetta.net/catalogPIP

tModelKeys
Registry Operation

- Peer nodes (Web sites)
- Companies register with any node
- Registrations replicated on a daily basis
- Complete set of “registered” records available at all nodes
- Common set of SOAP APIs supported by all nodes
- Compliance enforced by business contract
Registry Access Using SOAP

User
- UDDI SOAP Request
- UDDI SOAP Response

UDDI Registry Node
- HTTP Server
- SOAP Processor
- UDDI Registry Service
- B2B Directory

Create, View, Update, and Delete registrations

Implementation-neutral
UDDI Data Model

- UDDI includes an XML Schema that describes four core types of information:
  - businessEntity
    - Information about the actual business, e.g. business name, contact info etc.
  - businessService
    - Descriptive information about a particular service
  - bindingTemplate
    - Technical information about a service entry point and construction specifications
  - tModel (Technical Model)
    - Descriptions of specifications for services or taxonomies. Basis for technical fingerprints.
Core UDDI Entities

- businessEntity
  - bindingTemplate
  - businessService
    - bindingTemplate
    - businessService

- Interface tModel
  - Interface tModel
<Business>
  <businessEntity>
    :
    :
  </businessEntity>
</Business>

<Service>
  <businessService>
    <bindingTemplate>
      :
      :
    </bindingTemplate>
  </businessService>
</Service>

<TModel>
  <tModel>
    :
    :
  </tModel>
</TModel>
A. businessEntity

- businessEntity element includes info about the actual business
  - Business name, description, contact info such as address, phone, contact person, etc.

- Each business will receive a unique businessKey value when registering with a UDDI server
  - e.g. businessKey of Microsoft in its UDDI server: 0076b468-eb27-42e5-ac09-9955cff462a3

- The key is used to tie a business to its published services
businessEntity Structure

```
<businessEntity>
  <uddi:discoveryURLs />
  <uddi:name />
    <uddi:description />
  <uddi:contacts />
    <uddi:businessServices />
      <uddi:identifierBag />
    <uddi:categoryBag />
    <dsig:Signature />
</businessEntity>
```
• Can also include other unique value(s) in identifierBag that identifies the company
  – UDDI supports Dun & Bradstreet’s Data Universal System Number Identification System (DUNS) Numbers and Thomas Register Supplier Identifier Code System
  – e.g. Microsoft’s DUNS No: 08-146-6849
  – A D&B® (DUNS) Number is “a unique nine-digit sequence recognized as the universal standard for identifying and keeping track of over 100 million businesses worldwide.”

• Businesses can also register multiple business categories in categoryBag based on standard taxonomies, e.g.
  – NAICS (North American Industry Classification System) – defines codes for business and product categories, e.g., utilities, construction, and manufacturing
  – UNSPSC (Universal Standard Products and Service Code) – an open, global coding system for classifying products and services.”
<businessEntity
    businessKey="ba744ed0-3aaf-11d5-80dc-002035229c64">
    <name> XMethods </name>
    <description> … </description>
    <contacts>
        <contact> … </contact>
        <contact> … </contact>
    </contacts>
    <identifierBag> … </identifierBag>
    <categoryBag> … </categoryBag>
</businessEntity>
Examples of identifierBag and categoryBag contents (Microsoft)

```
<identifierBag>
  <keyedReference
    tModelKey="uuid:8609c81e-ee1f-4d5a-b202-3eb13ad01823"
    keyName="D-U-N-S" keyValue="08-146-6849" />
</identifierBag>

<categoryBag>
  <keyedReference
    tModelKey="uuid:c0b9fe13-179f-413d-8a5b-5004db8e5bb2"
    keyName="NAICS: Software Publisher"
    keyValue="51121" />
</categoryBag>
```
E.g. NAICS classification of Microsoft is “Software Publisher” with no. 51121.
B. businessService

- businessService element includes info about a single Web service or a group of related Web services

- Include the name, description and an optional list of bindingTemplates

- Like businessEntity, each businessService has a unique service key

- Should specify the businessKey to relate with the business that provides that service
<businessService
  serviceKey="d5921160-3e16-11d5-98bf-002035229c64"
  businessKey="ba744ed0-3aaf-11d5-80dc-002035229c64">
  <name>XMethods Delayed Stock Quotes</name>
  <description>...<\/description>
  <bindingTemplates>
    <bindingTemplate>
      :
    </bindingTemplate>
  </bindingTemplates>
</businessService>

To tie the service with the business

Typical contents of businessService element
C. bindingTemplate

- bindingTemplate element includes technical info about how and where to access a specific web service
- E.g. The Stock Quote Service is available via SOAP at http://services.xmethods.net:80/soap
- The serviceKey ties the bindingTemplate with the businessService “the Stock Quote Service”
bindingTemplates

- uddi:bindingTemplate
  - uddi:description
    - 0..∞
  - uddi:accessPoint
  - uddi:hostingRedirector
  - uddi:tModelInstanceDetails
  - uddi:categoryBag
  - dsign:Signature
    - 0..∞
<bindingTemplate
  serviceKey="d5921160-3e16-11d5-98bf-002035229c64"
  bindingKey="...">
  <description xml:lang="en">
  :
  </description>
  <accessPoint URLType="http">
    http://services.xmethods.net:80/soap
  </accessPoint>
  <tModelInstanceDetails>
  :
  </tModelInstanceDetails>
</bindingTemplate>

Typical contents of bindingTemplate element
- tModelInstanceDetails should further specify the key of the tModel used in this service

```xml
<tModelInstanceDetails>
  <tModelInstanceInfo
    tModelKey=
      "uuid:0e727db0-3e14-11d5-98bf-002035229c64"/>
</tModelInstanceDetails>
```
D. tModel

- tModels are primarily used to provide pointers to external technical specifications
- bindingTemplate only provides info about where to access the SOAP binding, but not how to interface with it
- tModel element fills this gap by providing a pointer to an external specification, such as WSDL
- In fact, tModels are not reserved to Web services
- tModels are used whenever it is necessary to point to any external specification, such as the D-U-N-S® no.
tModel Structure
tModel Structure

- Exactly one non-empty name
- Zero or more descriptions
- Zero or more overviewDocs (reference to remote instructions or descriptions related to the <tModel> and an <overviewURL> element.

- identifierBag
  - Contains tModelKey which uniquely identifies tModel
  - Other logical identifiers

- categoryBag
  - list of categories that describe specific aspects of the tModel
<tModel>
  tModelKey=
  "uuid:0e727db0-3e14-11d5-98bf-002035229c64"
  ...
  <description xml:lang="en">
    Simple stock quote interface
  </description>
  <overviewURL>
    http://www.xmethods.net/tmodels/
    SimpleStockQuote.wsdll
  </overviewURL>
</tModel>

A pointer points to the actual WSDL file
Define the category of this tModel
Categorization scheme: uddi-org:types
Category: wsdlSpec
A tModel typically represents a description of an Interface and can be referenced as part of the binding and instance information for a service.

A categorization scheme is a predefined set of categories, derived from an internal or external hierarchy, that can be used to classify a tModel.

**Categorizations**
- Categorization Scheme: uddi-org:types
- Key Name: uddi-org:types
- Key Value: wsdlSpec
- 1 record(s) found.
UDDI API

• There are two ways to search or publish a business/service
  – Using the Web pages provided by the UDDI implementation (UDDI cloud service), such as uddi.microsoft.com
    ⇒ Need human intervention
  – Using the APIs provided by UDDI
    ⇒ Can be made automatic by calling the APIs with computer programs

• UDDI APIs can be divided into two parts:
  – Inquiry APIs
  – Publishing APIs
<table>
<thead>
<tr>
<th>Inquiry Operations</th>
<th>Publishing Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find things</td>
<td>Save things</td>
</tr>
<tr>
<td>find_business</td>
<td>save_business</td>
</tr>
<tr>
<td>find_service</td>
<td>save_service</td>
</tr>
<tr>
<td>find_binding</td>
<td>save_binding</td>
</tr>
<tr>
<td>find_tModel</td>
<td>save_tModel</td>
</tr>
<tr>
<td></td>
<td>*****</td>
</tr>
<tr>
<td>Get details about things</td>
<td>Delete things</td>
</tr>
<tr>
<td>get_businessDetail</td>
<td>delete_business</td>
</tr>
<tr>
<td>get_serviceDetail</td>
<td>delete_service</td>
</tr>
<tr>
<td>get_bindingDetail</td>
<td>delete_binding</td>
</tr>
<tr>
<td>get_tModelDetail</td>
<td>delete_tModel</td>
</tr>
<tr>
<td>get_registeredInfo</td>
<td></td>
</tr>
<tr>
<td>get_registeredInfo</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Security</td>
</tr>
<tr>
<td></td>
<td>get_authToken, discard_authToken</td>
</tr>
</tbody>
</table>

Examples of Inquiry and Publishing APIs
UDDI Client

UDDI SOAP Request

UDDI SOAP Response

Wrapped by SOAP envelope and sent thru HTTP

UDDI Registry Node

HTTP Server

SOAP Server

Process UDDI API request

Registry
Example: find_business

Wrapped by a SOAP envelope

<envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <body>
    <find_business generic="1.0"
                    xmlns="urn:uddi-org:api">
      <name>XMethods</name>
    </find_business>
  </body>
</envelope>

Default namespace
Based on SOAP 1.1

Using UDDI 1.0

UDDI API find-business

Look for XMethod (ignore case, perform left-to-right lexical search)
A full record is shown, not partial

Business info about XMethod

Indicate the service(s) provided by XMethod

Response from Microsoft UDDI Registry
Example: get_businessDetail

```
<envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/">
  <body>
    <get_businessDetail generic="1.0"
      xmlns="urn:uddi-org:api">
      <businessKey>
        ba744ed0-3aaf-11d5-80dc-002035229c64
      </businessKey>
    </get_businessDetail>
  </body>
</envelope>
```

To query for the complete businessEntity record based on its key
Response from Microsoft UDDI Registry
Return a businessEntity record
An example of Publication, Inquiry and Invocation Process

Step 1

Company ABC

Publication

UDDI Registry

ABC Business
ABC Service
ABC tModel
• The objective of publishing business is to allow the clients to know the details of the business
  – such as the name of the company, the contact person, address and phone number etc.
• By publishing the service, the clients would know where and how to contact the service provider
  – such as the access point (or URL) of the service, transport protocol used (HTTP, SMTP, FTP or else)
• Publishing the tModel allows the clients to invoke the service provided by the business
  – based on the WSDL document of the service
Step 2

UDDI Registry

Company XYZ

find_business (ABC)

ABC’s

businessKey = ba744ed0-3aaf-11d5-80dc-002035229c64

serviceKey = d5921160-3e16-11d5-98bf-002035229c64
Step 3

Company XYZ

get_serviceDetail
(ABC’s serviceKey)

ABC’s service
accessPoint = http://services.xmethods.net:80/soap

tModelKey = uuid:0e727db0-3e14-11d5-98bf-002035229c64
Step 4

UDDI Registry

get_tModelDetail(tModelKey)

Company XYZ

tModel’s overviewURL = http://www.xmethods.net/tmodels/SimpleStock Quote.wsdl
Step 5

Company ABC

GLUE’s wsdl2java or IBM’s wsdlreader (access point and http://www.xmethods.net/tmodels/SimpleStockQuote.wsdl)

Company XYZ

WSDL document

Helper files

GLUE approach

ASSUME GLUE is being used

Reference: http://www.oreilly.com/catalog/webservices/chapter/ch06.html
Step 6

Based on the helper files, invoke the service

Company XYZ

Company ABC

ASSUME GLUE is being used
References

• Basic Information on UDDI
  – http://www.uddi.org/faq.html
    information on the basics of UDDI
  – http://uddi.microsoft.com
  – http://uddi.ibm.com
    UDDI Business Registry Nodes

• More Information on UDDI
  – http://www.uddi.org/whitepapers.html
    More technical information and UDDI
    Specifications
  – Ethan Cerami, Web Services Essentials, O'Reilly
    Sample chapter on WSDL and GLUE stuff