UDDI

Universal Description, Discovery and Integration protocol

Elizabeth Fischer
History of UDDI

- First specification proposed in 2000 by IBM, Ariba and Microsoft
- Most current version of specification v3.02 in September 2005
- Control of UDDI given to OASIS – info found in uddi.org
Purpose of UDDI

“A UDDI registry, either for use in the public domain or behind the firewall, offers a standard mechanism to classify, catalog and manage Web services, so that they can be discovered and consumed. “

UDDI V3.0.2 specification
Basic goals of UDDI

- Framework for describing and discovering business services, and service providers
- Defines data structures and APIs for publishing services descriptions to the registry and querying the registry
- Support developers in finding information about services
- Determine the security and transport protocols supported by a given Web service
- Support looking for services based on a general keyword
Ways to Use a UDDI registry

- **White Pages**
  - Obtaining listings of organizations, contact info, list of general services provided

- **Yellow Pages**
  - Look up information via standardized or user-defined taxonomies. UDDI supports standardized and user-defined classifications

- **Green pages**
  - Full descriptions of individual web services. Provided by pointers to service descriptions, which are usually stored outside of the registry
Structure of UDDI

- XML documents
- APIs are specified in WSDL with SOAP binding
- UDDI registry itself can be accessed as a web service
- Supports four core data structures:
  - businessEntity
  - businessService
  - bindingTemplate
  - tModel
Relationship of UDDI Core Data Types

- **businessEntity**: Information about the party who publishes information about a service.
  - businessEntities contain businessServices

- **businessService**: Descriptive information about a particular family of technical services.
  - businessServices contain bindingTemplates

- **bindingTemplate**: Technical information about a service entry point and implementation specs

- **tModel**: Descriptions of specifications for services or value sets. Basis for technical fingerprints.
  - bindingTemplates contain references to tModels. These references designate the interface specifications for a service.
businessEntity fields
<identifierBag>
  <keyedReference tModelKey="uddi:uddi.org:ubr:identifier:dnb.com:d-u-n-s"
                keyName="SAP AG"
                keyValue="31-626-8655" />
</identifierBag>
businessService fields
Each businessService structure represents a logical grouping of Web services. At the service level, there is still no technical information provided about those services; rather, this structure allows the ability to assemble a set of services under a common rubric. Each businessService is the logical child of a single businessEntity. Each businessService contains descriptive information – again, names, descriptions and classification information -- outlining the purpose of the individual Web services found within it. For example, a businessService structure could contain a set of Purchase Order Web services (submission, confirmation and notification) that are provided by a business.
bindingTemplate fields
bindingTemplate

每个bindingTemplate结构代表一个单独的Web服务。与businessService和businessEntity结构不同，这些结构旨在提供关于提供者和服务的辅助信息，bindingTemplate提供的是应用程序用于绑定和服务交互所需的技术信息。它必须包含给定服务的访问点，或者指向访问点的间接机制。

每个bindingTemplate是businessService的子结构。包含的父结构可以装饰有元数据，这些元数据可以发现bindingTemplate，根据一组参数和标准。

每个bindingTemplate是businessService的子结构。包含的父结构，可以被装饰有元数据，这些元数据可以发现bindingTemplate，根据一组参数和标准。
tModel fields
tModel describing the UDDI API

<tModel tModelKey="uddi:uddi.org:v3_publication">
  <name>uddi-org:publication_v3</name>
  <description>UDDI Publication API V3.0</description>
  <overviewDoc>
    <overviewURL useType="wsdlInterface">
      http://uddi.org/wSDL/uddi_api_v3_binding.wSDL#UDDI_Publication_SoapBinding</overviewURL>
    </overviewDoc>
    </overviewDoc>
    <overviewDoc>
      <overviewURL useType="text">
        http://uddi.org/pubs/uddi_v3.htm#PubV3
      </overviewURL>
    </overviewDoc>
    <categoryBag>
      <keyedReference keyName="uddi-org:types:wSDL"
        keyValue="wSDLSpec"
        tModelKey="uddi:uddi.org:categorization:types" />
    </categoryBag>
  </tModel>
... The UDDI information model is based on this notion of shared specifications and uses tModels to engender this behavior. For this reason, tModels exist outside the parent-child containment relationships between the businessEntity, businessService and bindingTemplate structures.

Each distinct specification, transport, protocol or namespace is represented by a tModel. ... To describe a Web service that conforms to a particular set of specifications, transports, and protocols, references to the tModels that represent these concepts are placed in the bindingTemplate. In such a way, tModels can be re-used by multiple bindingTemplates. The bindingTemplates that refer to precisely the same set of tModels are said to have the same "technical fingerprint" and are of the same type. In this way, tModels can be used to promote the interoperability between software systems.
Data encoded in tModels

- Transport and protocol definitions such as HTTP and SMTP.
- Value sets including identifier systems, categorization systems and namespaces. Structured categorizations using multiple value sets called "categorization groups."
- Postal address formats.
- Find qualifiers used to modify the behavior of the UDDI find_xx APIs.
- Use type attributes that specify the kind of resource being referred to by a URI reference.
Registering tModels

- UDDI.org will publish them on their members section.
- Registration is “limited to tModels that represent a well-known concept and/or are owned by a well-known standards group, an industry vertical or a consortium”.
- Must conform to UDDI best practices for coding.
- See currently registered tModels [here](#).
UDDI Registry keys

- Each entity is assigned a unique key
- V3 allows keys to be defined in a way that they are unique across registries
- Now URI-based, patterned on DNS names
- **UDDI key for UDDI API itself is**
  - uddi:uddi.org:categorization:types
UDDI Registry API

Six separate APIs that can be used by: service providers, requesters and other registries

- UDDI Inquiry API
- UDDI Publishers API
- UDDI Security API
- UDDI Custody and Ownership Transfer API
- UDI Subscription API
- UDI Replication API
UDDI Inquiry API

Intended for developers looking to locate services, and for clients at run-time for dynamic binding

- find_business, find_service, find_binding, find_tModel
- get_businessDetail, get_serviceDetail, get_bindingDetail, get_tModelDetail
Purposes for UDDI Registries

- Public implementation – UDDI Universal Business Registry – now disbanded - see here
- Stated purpose now seems to be within a business infrastructure See vendor toolkits available