Lesson 7 – Directory services (Part II)

Service Oriented Architectures Security

Module 1 - Basic technologies

Unit 4 – UDDI

Ernesto Damiani

Università di Milano

Binding template (1)

- A binding template contains the technical information associated to a particular service. It contains the following information:
  - bindingKey
  - serviceKey
  - description
  - accessPoint: the network address of the service being provided
  - typically a URL but it can be anything as this field is a string: e.g., an email address or even a phone
  - tModels: a list of entries corresponding to tModels associated with this particular binding. The list includes references to the tModels,
  - documents describing these tModels, short descriptions, etc.
  - categoryBag: additional information about the service and its binding (e.g., whether it is a test binding, it is on production, etc.)

Binding template (2)

- A businessService can have several binding templates, but a bindingTemplate has only one businessService
- The binding template can be best seen as a folder where all the technical information of a service is put

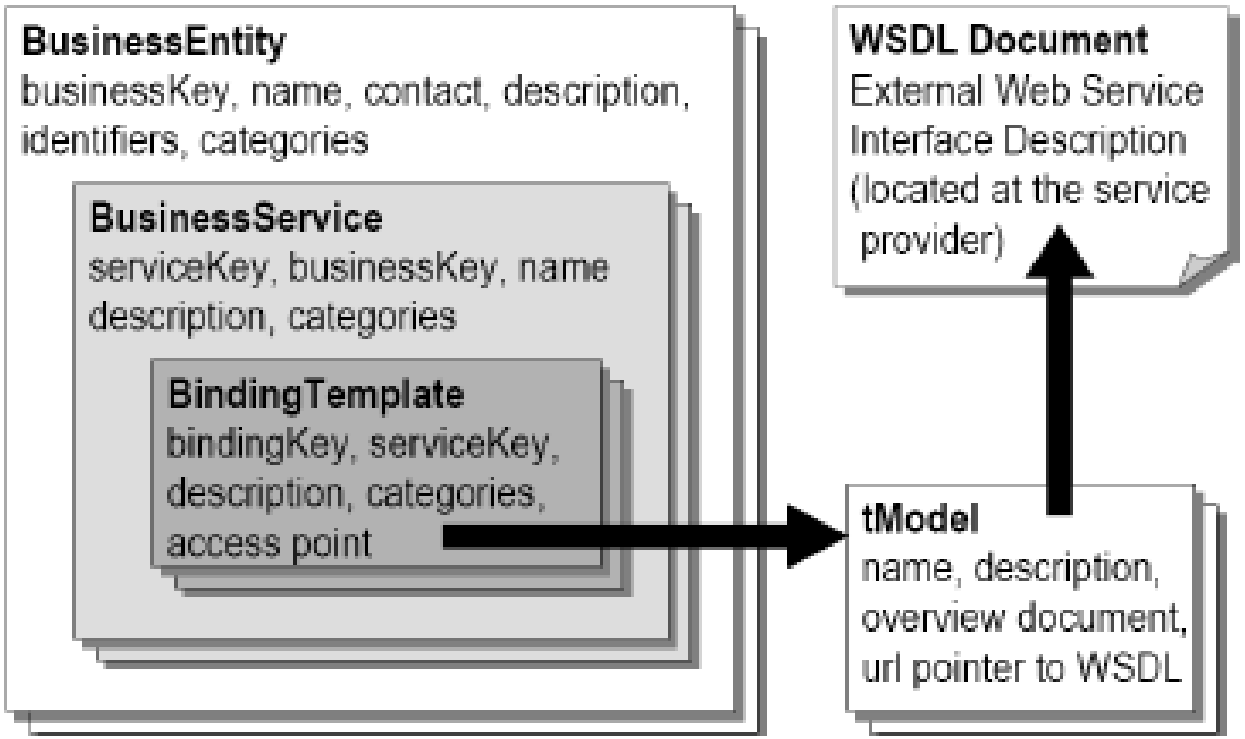
tModel (1)

- A tModel is a generic container of information where designers can write any technical information associated to the use of a Web service:
  - the actual interface and protocol used, including a pointer to the WSDL description
  - description of the business protocol and conversations supported by the service
  - “any concept that is not better represented by one of the other UDDI data structures”

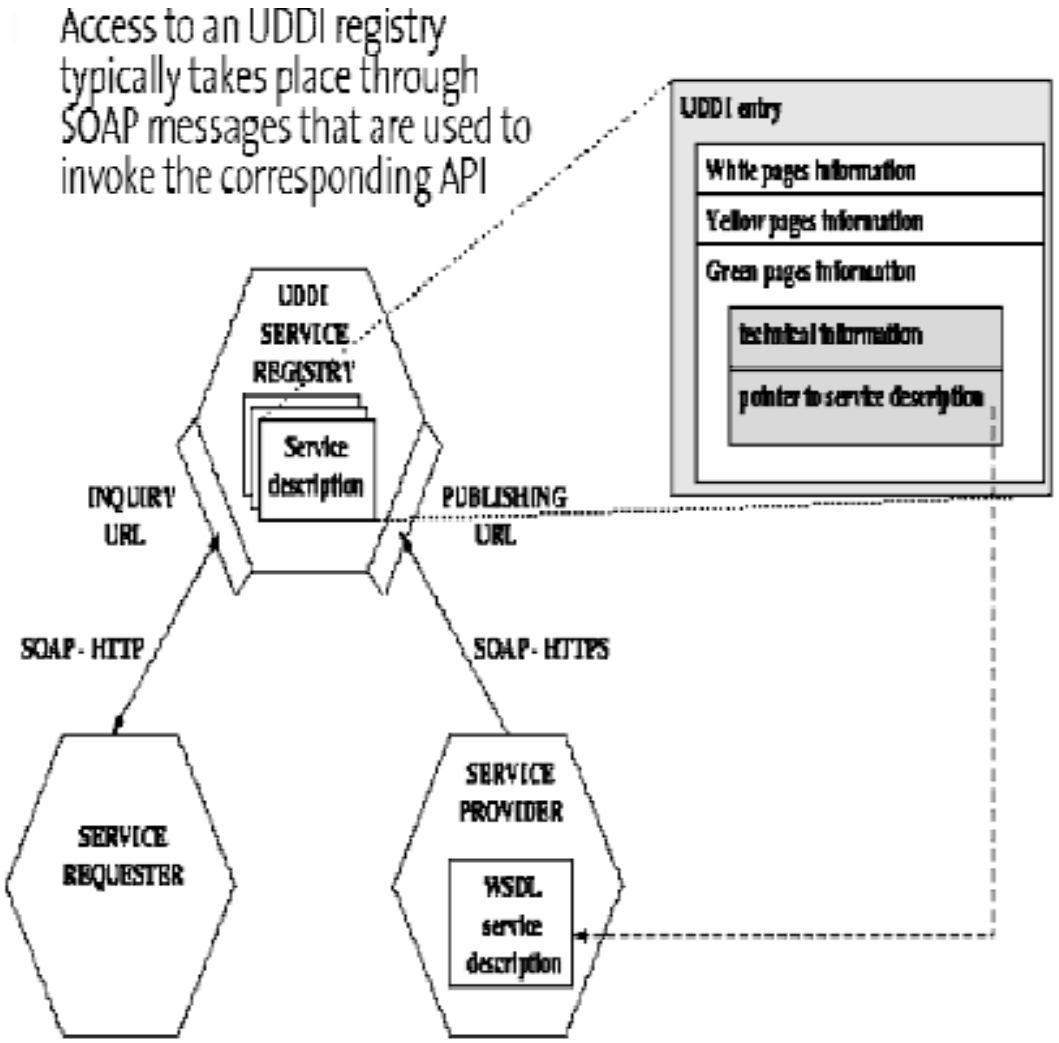
tModel (2)

- A tModel is a document with a short description of the technical information and a pointer to the actual information. It contains:
  - tModelKey
  - Name & Description
  - overviewDoc: (with an overview URL and useType that indicate where to find the information and its format, e.g., “text”or “wsdldescription”)
  - identifierBag& categoryBag
- A tModel can point to other tModels and eventually different forms of tModels will be standardized (tModel for WSDL services, tModels for EDI based services, etc.)

UDDI data model



Inquiry and publishing interface



UDDI interfaces (1)

- The UDDI specification provides a number of APIs (Application Program Interfaces) that provide access to an UDDI system:
  - UDDI Inquiry: to locate and find details about entries in an UDDI registry. Support a number of patterns (browsing, drill-down, invocation)
  - UDDI Publication: to publish and modify information in an UDDI registry. All operations in this API are atomic in the transactional sense
  - UDDI Security: for access control to the UDDI registry (token based)
  - UDDI Subscription: allows clients to subscribe to changes to information in the UDDI registry (the changes can be scoped in the subscription request)

UDDI interfaces (2)

- UDDI Replication: how to perform replication of information across nodes in an UDDI registry
- UDDI Custody and Ownership transfer: to change the owner (publisher) of information and ship custody from one node to another within an UDDI registry

• UDDI also provides a set of APIs for clients of an UDDI system:

- UDDI Subscription Listener: the client side of the subscription API
- UDDI Value Set: used to validate the information provided to an UDDI registry

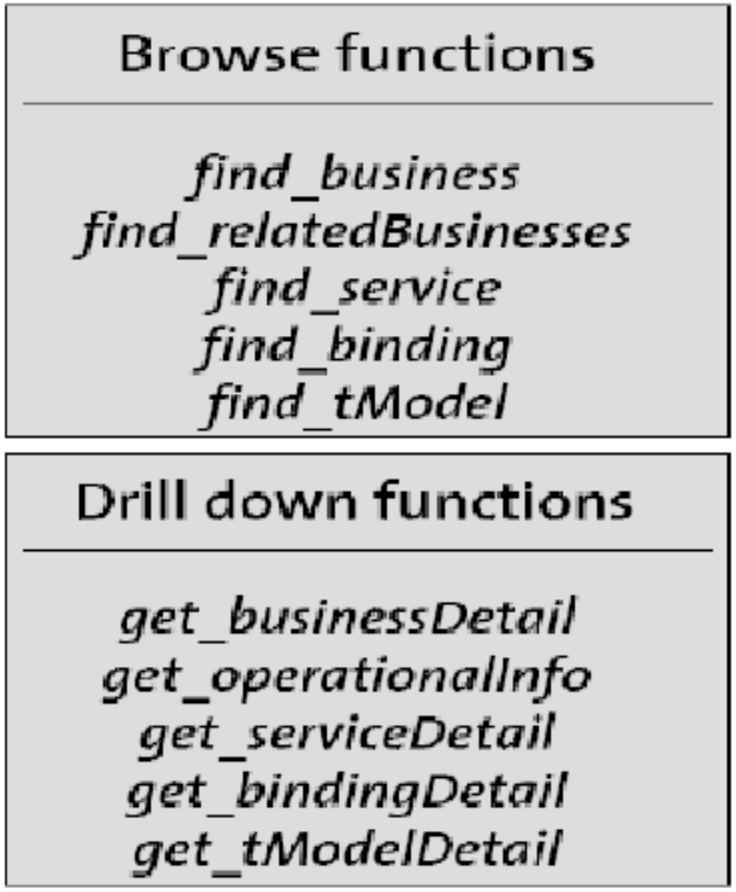
Inquiry API (1)

- Search and lookup entries in a registry
- This API is freely available, no client authentication is required
- Errors are reported as SOAP Faults
- Browse functions search the registry based on keywords and return summary lists with overview information (key, name and description) about matching businesses or services

Inquiry API (2)

- Find qualifiers are used to sort the results and to control the keyword matching: toggle between AND/OR, case sensitive/insensitive, use of wildcards and categories
- To minimize the number of requests, find queries can be nested

Inquiry API (3)



Security API (1)

- Publish, update and delete information contained in an UDDI registry
- The publishing API requires user authentication using a session token and typically uses SOAP over HTTPS
- The registry performs access control for all publishing functions: information about the entries can only be edited by the owner

Security API (2)

- Category information and keyed references associated to the entries are validated before accepting new information into the registry
- Deletion functions are used to remove entries identified by their key from the registry. Removing a business will remove all services associated with it

Security API (3)

UDDI summary (1)

- The UDDI specification is rather complete and encompasses many aspects of an UDDI registry from its use to its distribution across several nodes and the consistency of the data in a distributed registry
- Most UDDI registries are private and typically serve as the source of documentation for integration efforts based on Web services

UDDI summary (2)

• UDDI registries are not necessarily intended as the final repository of the information pertaining Web services. Even in the “universal” version of the repository, the idea is to standardize basic functions and then built proprietary tools that exploit the basic repository. That way it is possible to both tailor the design and maintain the necessary compatibility across repositories

UDDI summary (3)

• While being the most visible part of the efforts around Web services, UDDI is perhaps the least critical due to the complexities of B2B interactions (establishing trust, contracts, legal constrains and procedures, etc.). The ultimate goal is, of course, full automation, but until that happens a long list of problems need to be resolved and much more standardization is necessary

UDDI limitations (1)

- There were a few universal UDDI registries in operation (maintained by IBM, Microsoft, SAP, etc)
- These registries were very visible and often the first thing one saw of Web services
- Most of the entries in them did not work or did not correspond to any real service

UDDI limitations (2)

- This has been a source of criticism to Web services in general. The criticism has not been entirely undeserved but it is often misguided: what was there to criticize was not UDDI itself but the use that was been made of it and the hype
- Supporting infrastructure for Web services in well defined and constrained environments (i.e., without public access and where there is a context that provides the missing information)

UDDI limitations (3)

- Most of the UDDI registries in place today are private registries operating inside companies (recall that the widest use of Web services today is for conventional EAI) or maintained by a set of companies in a private manner
- UDDI has now become the accepted way to document Web services and supply the information missing in WSDL descriptions

Public UDDI registries (1)

- Former UBR (UDDI Business Registry) nodes:
  - IBM
    - Homepage: http://uddi.ibm.com/
    - Inquiry API: http://uddi.ibm.com/ubr/inquiryapi
    - Publish API: https://uddi.ibm.com/ubr/publishapi
  - Microsoft
    - Homepage: http://uddi.microsoft.com/
    - Inquiry API: http://uddi.microsoft.com/inquire
    - Publish API : https://uddi.microsoft.com/publish
- The public UDDI Business registries provided by IBM, Microsoft and SAP have been discontinued since January 2006 (http://uddi.microsoft.com/about/FAQshutdown.htm

Public UDDI registries (2)

- Since their launch in Sept. 2000, they accumulated over 50‘000 service registration entries
  - SAP
    - Homepage: http://uddi.sap.com/
    - Inquiry API : http://uddi.sap.com/uddi/api/inquiry
    - Publish API : https://uddi.sap.com/uddi/api/publish
  - NTT
    - Homepage: http://www.ntt.com/uddi/
    - Inquiry API : http://www.uddi.ne.jp/ubr/inquiryapi
    - Publish API : https://www.uddi.ne.jp/ubr/publishapi