



7363 - Web-basierte Anwendungen

Eine Vertiefungsveranstaltung
mit Schwerpunkt auf XML-Technologien



UDDI

Universal **D**escription, **D**iscovery, **I**ntegration



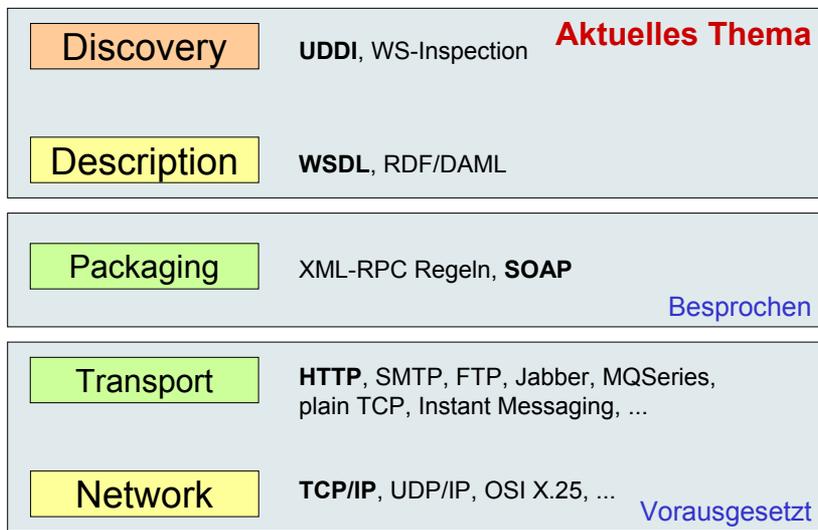
UDDI-Übersicht



SOAP



Der Technologie-Stack von Web Services:





- Warum UDDI?
 - Zum Auffinden von Anbietern
 - Zum Finden von Diensten der Anbieter
 - Zur Beschreibung dieser Dienste
 - verbal, für menschliche Leser
 - technisch, für Entwickler bzw. für Anwendungen
- Analoga im Telefonnetz
 - "Gelbe Seiten":
 - Suche Anbieter aus gegebener Branche
 - "Weiße Seiten":
 - Suche nach Kontaktdaten eines gegebenen Anbieters
 - "Grüne Seiten" (hier nicht verbreitet):
 - Techn. Details



- Status
 - Industriestandard, nicht W3C!
 - Zunächst UDDI *group*, später: OASIS
 - *Releases*
 - 2001: UDDI 1.0 (Anfänge seit 1999)
 - 2002-07-19: UDDI 2.04 API Spec. & 2.03 *Data Structure Ref.*
 - 2003-10-14: UDDI 3.01 (Unterschiede: *Evolution-Whitepaper*)
 - 2005-02-03: UDDI V3.0.2 als OASIS-Standard ratifiziert
- Quellen
 - <http://www.uddi.org>
 - UDDI XML Schema:
 - Stand 2001: http://www.uddi.org/schema/uddi_v1.xsd
 - Stand 1999: http://www.uddi.org/schema/uddi_1.xsd
 - Namensraum-URIs: urn:uddi-org:api, urn:uddi-org:publication



- Entwicklung von UDDI
 - "From e-business directory to web services infrastructure"
 - Service-oriented architecture (SOA)
 - Zunächst "*Universal Business Registry (UBR)*"
 - Fokus erst später auf Technik & Integrationsaspekte gerichtet
 - [Quelle: *Evolution-Whitepaper*]
- Treibende Kräfte für Weiterentwicklung
 - Integration mit anderen sich entwickelnden Standards
 - XML Schema, Web Services, WSDL, XML Signature
 - Flexiblere Schlüsselvergabe
 - Lokale Vergabe, dennoch globale Eindeutigkeit
 - Übergang zu URN-artigen Schlüssel (Präfix "uddi") statt UUID
 - Replikation zwischen *Registries per Web Service*
 - *Subscription*
 - Beeinflussung durch neue Anforderungen, insb. ebXML „RS/RIM“



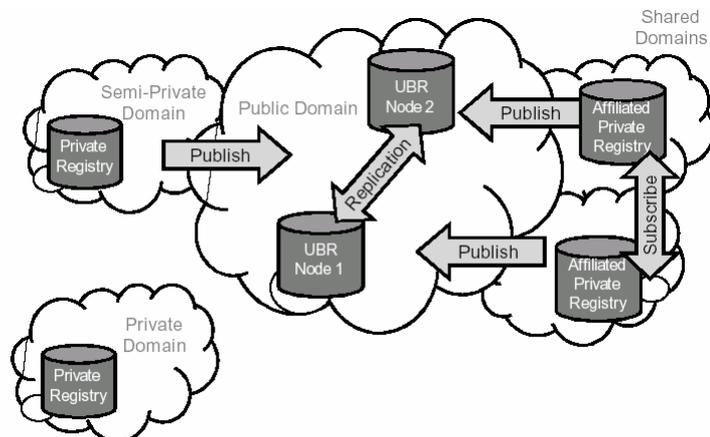
- UBR-Inhalte
 - "Gelbe Seiten"
 - Sortierung nach Branchen
 - Hilfe zur Suche eines *Service Providers*
 - "Weiße Seiten"
 - Nachschlagen der Kontaktdetails eines bereits bekannten *Providers*
 - "Grüne Seiten"
 - Technische Informationen über angebotene Services
 - Service-Typen
 - tModels („*technical models*“)



- **Deployment**
 - **Public**
 - *Service cloud, Global Business Registry,*
 - **UDDI Public Registry operators**
 - Ursprünglich Microsoft, IBM, Ariba;
 - zwischenzeitlich HP statt Ariba, nun: SAP und NTT Com
 - **Private, Mischformen**
 - *Closed user groups*
 - *Public enquiry, private publication*
 - *Value added services*
 - **Analogie:**
 - *Intranet vs. Extranet vs. Internet*
- **Neu: subscribe**
 - Bezug von Änderungsmitteilungen und neuen Einträgen



- **Interaktionen zwischen registries**



Quelle: *The Evolution of UDDI. Whitepaper by The Stencil Group*



- UDDI innerhalb größerer Unternehmen
 - als *service registry* konzernweiter EAI-Projekte
 - EAI = *Enterprise Application Integration*
 - zur Dokumentation
 - zur Steuerung des Wechsels
 - Test- / Konsolidierungs- / Produktions-Phase
 - zur erleichterten Koordination globaler Teams
 - zur Integration über Systemgrenzen hinweg
 - zur kontrollierten Freigabe technischer Einzelheiten
 - als Ausgangsbasis für firmenübergreifende Kooperationen



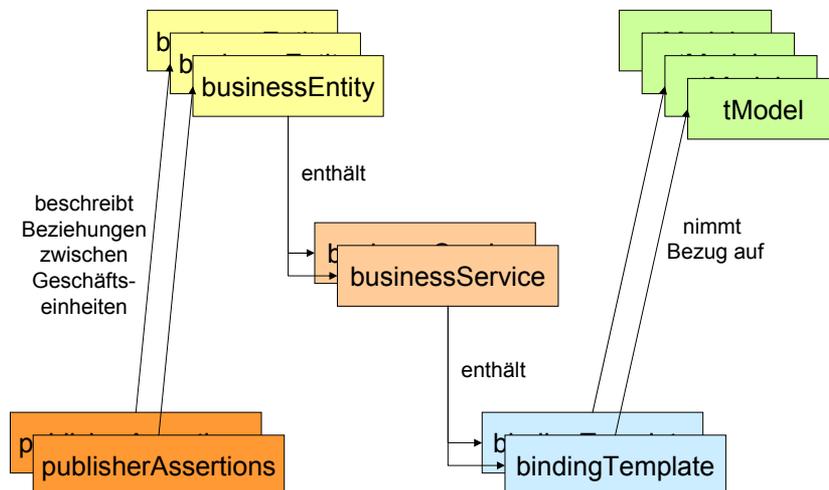
UDDI-Anatomie



- "Anatomie" von UDDI Registry-Inhalten
 - **Business Entities**
 - Beschreibung von Unternehmen
 - Entspricht "white pages"-Einträgen
 - **Business Services**
 - Die von einer Firma angebotenen Dienste
 - Nicht auf *Web Services* beschränkt!
 - **Binding Templates**
 - Technische Information ("*green pages*") über einen *Web Service*
 - Die zur Verbindung und Aufruf notwendigen Detailangaben
 - **TModels**
 - "*Technical*" models, ein Sammelbegriff für diverse Spezifikationen
 - **Publisher Assertions** (ab V. 2.0)
 - Zur Veröffentlichung von Geschäftsbeziehungen



Beziehungen zwischen UDDI-Elementen





- **businessEntity**

- enthält folgende Elemente

- discoveryURLs

- **name**

← Muss-Element

- description

- contacts

- **businessServices**

← Einzelheiten folgen

- identifierBag

- categoryBag

- enthält folgende Attribute

- **businessKey**

← Pflicht-Attribut

- operator

- authorizedName



- **discoveryURLs**

- Container-Element, enthält "discoveryURL"-Elemente

- Zur Angabe von (alternativen) URLs zur formalen Beschreibung.

Beispiel:

```
<discoveryURLs>
  <discoveryURL useType="businessEntity">
    http://uddi.sap.com/UDDI/discovery/businessEntity/
    a694dcd4-9d88-11d6-91b6-0003479a7335
  </discoveryURL>
  <discoveryURL useType="homepage">
    http://www.sap.com/
  </discoveryURL>
</discoveryURLs>
```

- **name**

- Name des Unternehmens. Beispiel:

```
<name xml:lang="en">SAP AG</name>
```

- **description**

- Freitext zur Beschreibung des Unternehmens



- **contacts**
 - Container-Element, enthält "contact"-Elemente
 - Zur Angabe üblicher Kontaktdaten wie Namen von Personen, Telefon- und Faxnummern, Anschrift.
- **businessServices**
 - Liste der angebotenen Dienste, siehe unten
- **identifierBag**
 - Liste von Name/Wert-Paaren zur Identifizierung des Unternehmens
 - gemäß bestimmter Standards wie US Tax Code Id, D-U-N-S, (hoffentlich auch) GLN, BLZ/BIC
- **categoryBag**
 - Analog, zur Identifizierung der Branche
 - gemäß (hierarchischer) Standards wie UNSPSC [, eCI@ss, ETIM]



- **Attribute von „businessEntity“:**
 - **businessKey**
 - Eindeutiger Schlüssel zur Identifizierung dieses Eintrags
 - Siehe Folien zum Stichwort "UUID"
 - **operator**
 - URL des UDDI/UBR *operators*
 - z.B.: "www.ibm.com/services/uddi"
 - **authorizedName**
 - Name bzw. Code der Person, die den Eintrag publizierte



- **DCE UUID (Universal Unique Identifier, auch: GUID)**
 - Wird von einem UDDI *operator* vergeben
 - Ist garantiert eindeutig, hier: im gesamten öffentlichen UDDI *registry*
 - Beispiel (SAP's *business key* incl. Präfix):
 - `uddi : a694dcd4-9d88-11d6-91b6-0003479a7335`

Weiterführende Quellen zu UUID:

- [1] Zahn, L., Dineen, T. and P. Leach, "Network Computing Architecture", ISBN 0-13-611674-4, January 1990.
- [2] "DCE: Remote Procedure Call", Open Group CAE Specification C309, ISBN 1-85912-041-5, August 1994.
- [3] <http://www.ietf.org/internet-drafts/draft-mealling-uuid-urn-03.txt>
- [4] <http://ftp.ics.uci.edu/pub/ietf/webdav/uuid-guid/draft-leach-uuids-guids-01.txt>
- [5] ISO/IEC 11578:1996



- **DCE UUID: Nachteile im Kontext von UDDI**
 - Kollision mit URI/URN-Konzept
 - UUIDs sind bestenfalls als Teil eines URI verwendbar, während sie Funktionen eines kompletten URI übernehmen
 - Global eindeutige Vergabe wird auch von URIs sichergestellt
 - Nicht-sprechende, kryptische Codes
 - Siehe Beispiel
 - Vergabe kann nicht durch Anwender selbst geschehen
 - Bei URIs ist das dagegen (*domain-intern*) selbstverständlich
- **Ausweg: "Publisher Assigned Keys"**
 - URN-Syntax für UDDI-Schlüssel, verfügbar ab UDDI 3.0
 - Beispiel: `uddi : somecompany . com : serviceName`



- **businessService**

- enthält folgende Elemente

- **name**
- description
- **bindingTemplates**
- categoryBag

- enthält folgende Attribute

- **serviceKey**
- businessKey



- **name**

- Name des Dienstes (Klartext, kurz)

- **description**

- Freitext zur Beschreibung des Dienstes

- **bindingTemplates**

- Liste der angebotenen Dienste, siehe Besprechung zu "**bindingTemplate**"

- **categoryBag**

- Vergleiche businessEntity/categoryBag

- Attribute:

- **businessKey:** Stellt Bezug her zu businessEntity-Eintrag
- **serviceKey:** Zur eindeutigen Identifizierung dieses Dienstes



- **bindingTemplate**
 - enthält folgende Elemente
 - description
 - accessPoint | hostingRedirector
 - **tModellInstanceDetails**
 - enthält folgende Attribute
 - **bindingKey**
 - serviceKey



- **description**
 - Freitext zur Beschreibung der Schablone
- **accessPoint**
 - Enthält Kontakt-Details, meist in URL-Form
 - Attribut urlType: `http`, `https`, `ftp`, `fax`, `phone`, `mailto`
- **hostingRedirector**
 - Verweis auf ein anderes *binding template* falls dieses leer
- **tModellInstanceDetails**
 - i.w. Liste der involvierten **tModel**-Einträge, siehe dort
- Attribute:
 - **bindingKey:** Zur eindeutigen Identifizierung dieser Schablone
 - **serviceKey:** Stellt Bezug her zum businessService-Eintrag



- **tModel**

- enthält folgende Elemente

- **name**
- description
- **overviewDoc**
- identifierBag
- categoryBag

- enthält folgende Attribute

- **tModelKey**
- operator
- authorizedName



- **name**

- Name des Modells

- **description**

- Freitext zur Beschreibung des Modells

- **overviewDoc**

- Eine Referenz (etwa: URL) zu Hintergrundinformation
- Hier kann insb. ein WSDL-Dokument referenziert werden!

- **identifierBag, categoryBag**

- Vergleiche "businessEntity"

- Attribute:

- **tModelKey:** Zur eindeutigen Identifizierung dieses Modells
- operator, authorizedName: Vgl. "businessEntity"



- **Bemerkungen zu tModel-Einträgen**
 - Die Bedeutung der tModels ist bewusst vage gehalten.
 - Sie bieten Raum für Hintergrundinformationen, die den jeweiligen Dienst erst im Detail erklären.
- **Beispiele:**
 - WSDL- und XML Schema-Angaben
 - Namensräume
 - Hintergründe zu verwendeten Identifizierungsstandards, Dokumentationen in HTML- oder PDF-Form, etc.
 - Hinweise auf Standard-Organisationen
- **Entfernte Analogie zu XML's NOTATION**
- **Auch tModel-Einträge werden klassifiziert und können analog zu businessEntity-Einträgen gesucht werden.**



- **Demo**
 - Suche nach Web Services mit Hilfe des UDDI Web Interfaces von SAP unter:
 - https://www001.sap-ag.de/~form/uddi_discover/prod
 - Stichwortsuche
 - "SAP", dann "SAP AG" (Haupteintrag),
 - ferner "SAP AG Business Registry Node"
 - Publish- und Query-Unterpunkt
 - Ziel: Vollständige businessEntity-Einträge!
 - "Browsing" / Suche nach Regionen
 - Was gibt's aus Hessen?



UDDI-APIs



UDDI



- **UDDI APIs**

- Publisher-Interface "**PublishSOAP**"
 - 16 Methoden
- Inquiry-Interface "**InquireSOAP**"
 - 10 Methoden
- Natürlich SOAP-Schnittstellen, beschrieben mit WSDL, basierend auf UDDI XML Schema
 - Dokumentenmodus!
- Ferner: Replikationsschnittstellen
- Benutzerverwaltung (lokal)



• PublishSOAP: Methoden

- zum Speichern
 - `save_business`, `save_service`, `save_binding`,
`save_tModel`
- zum Löschen
 - `delete_business`, `delete_service`, `delete_binding`,
`delete_tModel`
- für die *Account*-Verwaltung
 - `get_authToken`, `discard_authToken`;
`get_registeredInfo`
- für den Umgang mit *Assertions*
 - `add_publisherAssertions`,
`get_publisherAssertions`,
`set_publisherAssertions`,
`delete_publisherAssertions`,
`get_assertionStatusReport`



• PublishSOAP: Typischer Ablauf

- Einmalig: Bei einem der *operators* registrieren.
 - Alle Publikationen nur über diesen tätigen!
- Typische *session*
 - Login
 - *Session Token* holen, mit `get_authToken`
 - Verschiedene Aktivitäten wie speichern, ändern, löschen
 - Jeweils *Token* erforderlich
 - Logout
 - *Token* für ungültig erklären, mit `discard_authToken`
- Bem.:
 - Lesen darf jeder. Ähnlichkeit mit Veranstaltungs-Server!



• InquireSOAP: Methoden

- zur Suche ("*drill-down style*")
 - `find_business`, `find_service`, `find_binding`,
`find_tModel`
 - `find_relatedBusiness`

- für Detailabruf, auf Basis der jeweiligen Objekt-IDs
 - `get_businessDetail`, `get_serviceDetail`,
`get_bindingDetail`, `get_tModelDetail`
 - `get_businessDetailExt`



• InquireSOAP: Typischer Ablauf

- `find_business`
 - Passendes `businessEntity`
- `get_businessDetail`
oder gleich `get_serviceDetail`
 - Einzelheiten zum gewünschten WS
- `get_tModelDetails`
 - Weitere technische Einzelheiten

- Abruf der so gefundenen WSDL-Datei
- Aufbau der Schnittstelle gemäß WSDL-Beschreibung
- Aufruf des Dienstes!



- **UDDI-Toolkits** (offensichtlich Java-dominiert)
 - IBM: **UDDI4J** (<http://www.sourceforge.net/projects/uddi4j>)
 - Sowohl Client- als auch Server-Komponenten
 - Am häufigsten genannt, daher möglicherweise besonders ausgereift.
 - IBM Public License
 - SAP: SAP Web Application Server (**Web AS**)
 - Teil von NetWeaver (www.sap.com/solutions/netweaver)
 - Apache **jUDDI** ("Judy")
 - <http://www.apache.org/juddi/>
 - Weitere: Siehe <http://www.uddi.org/solutions.html>



WS-Inspection

Eine einfachere Alternative zu UDDI
bei reduzierten Anforderungen



- Ausgangspunkt
 - *WS Provider* und *WS Consumer* kennen sich bereits.
 - *WS Provider* möchte die Details zur Benutzung seiner Dienste veröffentlichen.
- Konvention (Beispiel)
 - Datei "inspection.wsil" im root-Verzeichnis des Web Servers des *Providers*:
 - <http://www.example.com/inspection.wsil>
 - Diese Datei enthält die Beschreibung aller Dienste dieses Anbieters.
- Nutzung
 - *Consumer* schaut die Details beim bereits bekannten *Provider* nach, konfiguriert & ruft auf.



- Herkunft der "*Web Services Inspection Language*"
 - IBM und Microsoft
- Aufbau
 - Dokumenten-Element: inspection
 - Direkte Unter-Elemente: abstract, link, service
 - Unterelemente von "service": abstract, description
 - Namensraum:
<http://schemas.xmlsoap.org/ws/2001/10/inspection/>
- Quellen
 - [1] <http://www-106.ibm.com/developerworks/webservices/library/ws-wsilspec.html>
 - [2] <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnglobspec/html/ws-inspection.asp>



```
<?xml version="1.0"?>
<inspection xmlns="http://schemas.xmlsoap.org/ws/2001/10/inspection/"
  xmlns:wsiluddi="http://schemas.xmlsoap.org/ws/2001/10/inspection/uddi/">

  <service>
    <abstract>A stock quote service with two descriptions</abstract>
    <description
      referencedNamespace="http://schemas.xmlsoap.org/wsdl/"
      location="http://example.com/stockquote.wsdl"/>
    <description referencedNamespace="urn:uddi-org:api">
      <wsiluddi:serviceDescription
        location="http://www.example.com/uddi/inquiryapi">
        <wsiluddi:serviceKey>4FA28580-5C39-11D5-9FCF-BB3200333F79
        </wsiluddi:serviceKey>
        </wsiluddi:serviceDescription>
      </description>
    </service>

    <service>
      <description
        referencedNamespace="http://schemas.xmlsoap.org/wsdl/"
        location="ftp://anotherexample.com/tools/calculator.wsdl"/>
    </service>

    <link referencedNamespace="http://schemas.xmlsoap.org/ws/2001/10/inspection/"
      location="http://example.com/moreservices.wsil"/>

  </inspection>
```