Hauptseminar Internetdienste
Prof. F. Schweiggert
Sommersemester 2004
Universität Ulm

Web Services

Boto Bako

Inhaltsverzeichnis

1. Einführung und Motivation

Web-Services sind angebotene Dienste über das World Wide Web. Klienten können diese Dienste nutzen, um Endbenutzer-Anwendungen zu realisieren, sie zu Geschäftstransaktionen zusammenzufassen oder um neue Web-Services anzubieten. Web-Services werden durch XML-basierte Standardschnittstellen angesprochen. Das Ziel ist Kommunikationautomatisierung über das Internet.

Die Vision von Web Services ist der Markt von Komponenten ([Hau02]). Dabei erstellen unabhängige Softwarefirmen Web-Servicekomponenten, die von verschiedenen Dienstanbietern angeboten werden. Eine Beispielanwendung wäre ein Onlineshop: Der Onlineshop wickelt die Bestellungen vom Kunden ab. Über ein Web-Services Registry wird nach geeigneten Großhändlern gesucht, um die Waren nachbestellen zu können. Dazu werden die Bestelldaten automatisch an die Web-Services der Großhändler weitergeleitet. Diese können ihrerseits andere Web-Services aufrufen, etwa um einen Transportunternehmen mit der Lieferung zu beauftragen. Genauso kann die Kreditkarten-Bezahlung über ein Web-Service eines Kreditinstitutes abgewickelt werden.

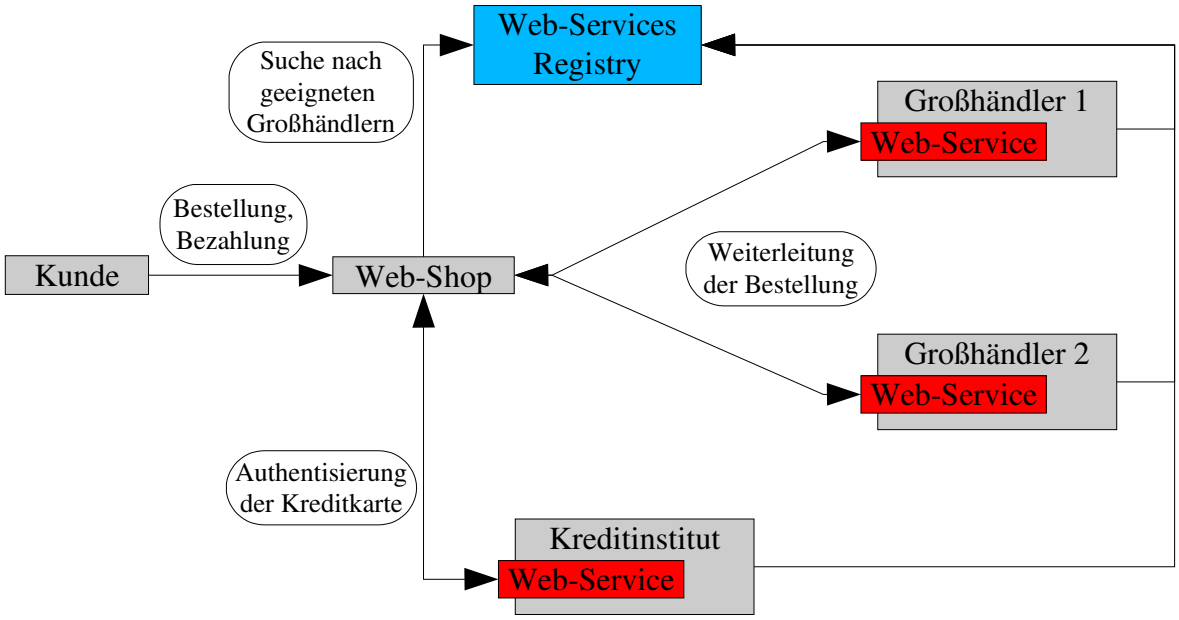

Abbildung 1: Beispielanwendung von Web-Services

Die Nachteile bisheriger Komponentenmodelle sind, dass sie unterschiedliche Protokolle für die Kommunikation benutzen und unterschiedliche Schnittstellenbeschreibungen anbieten. Durch Einhaltung verschiedener Standards sind Web-Services plattform- und sprachunabhängig. Web Services sind aber kein Ersatz bisheriger Komponentenmodelle wie Corba oder Enterprise Java Beans, sondern vielmehr eine Vereinheitlichung. Web Services werden mit Hilfe dieser Frameworks implementiert, wodurch die vorhandenen Technologien mit ihren Vorteilen genutzt werden können.

2. Verwendete Standards

Die vorher genannte Vereinheitlichung wird durch die Einhaltung folgender Standards erreicht ([Hau02]):

- Kommunikation über SOAP
  (*Simple Object Access Protocol*)
- Beschreibung von Diensten über WSDL
  (*Web Services Description Language*)
- Auffinden und Registrierung über UDDI
  (*Universal Service Description, Discovery, and Integration*)

Ein Web Service ist also ein Dienst, dessen Schnittstellen mit WSDL beschrieben, mit UDDI registriert und gefunden und mit SOAP angesprochen werden können. Der Zugang zu so einem Dienst erfolgt über das Web (SOAP basiert auf HTTP), unter Verwendung von standardisierten und verbreiteten Protokollen und Konzepte, wie HTTP und XML. Dabei zeichnen sich folgende Rollen in der Web-Services Architektur ab:

- *Service Registry* (Verzeichnisdienst)
  Verwaltet registrierte Dienste und ermöglicht die Suche nach speziellen Diensten.
- *Service Requestor* (Dienstnachfrager)
  Ein Klient oder ein anderer Web-Service, der ein Dienst in Anspruch nehmen will. Die Suchanfragen werden an den Verzeichnisdienst geschickt.
- *Service Provider* (Dienstanbieter)
  Der Anbieter der Dienstleistung. Ist zuständig für die Implementierung, Bereitstellung der Schnittstellen (WSDL) und Registrierung beim Verzeichnisdienst.

Einen Überblick über die Rollenverteilung und die eingesetzten Protokolle gibt folgende Abbildung ([Smi04]):

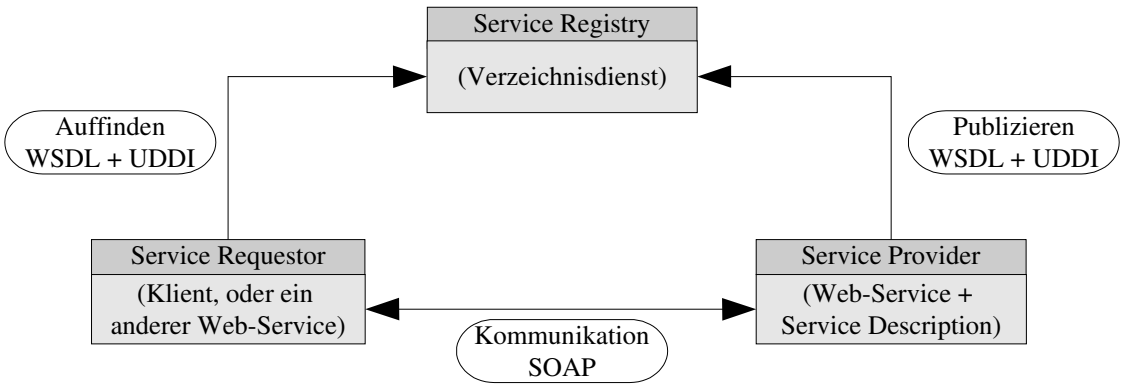

Abbildung 2: Rollen in Web-Services

2.1. *SOAP*

SOAP ist das Kommunikationsprotokoll für Web-Services. Es findet eine nachrichtenbasierte Übertragung statt, es lassen sich aber mittels Nachrichtenverbunde auch komplexere Kommunikationsparadigmen, wie Request-Reply-Protokolle oder Multicast-Protokolle, realisieren ([Hau02]). Die Nachrichten sind XML-Dokumente. Als Trägerprotokoll wird meist HTTP eingesetzt, kann aber auch über andere Protokolle abgewickelt werden.
Den Aufbau einer SOAP-Nachricht zeigt folgendes Bild:

```
<SOAP:Envelope xmlns:SOAP=  http://www.w3.org/2003/05/soap-envelope >

   <SOAP:Header>
      <signature>  ...  </signature>
   </SOAP:Header>

   <SOAP:Body>
      <a:withdraw xmlns:a="http://eineBank.com/">
         <a:account>1234</a:account>
         <a:amount>200.25</a:amount>
      </a:withdraw>
   </SOAP:Body>

</SOAP:Envelope>
```

Abbildung 3: Aufbau einer SOAP-Nachricht

- *Envelope*
  „Umschlag“d er Nachricht, eine XML-Datei mit XML-Namensraum für SOAP-Tags. Enthält Header und Body.
- *Header* (optional)
  Enthält anwendungsspezifische Daten, bspw. für Authentifizierung, Transaktionen und Routing. Zwischenknoten können den Header verändern.
- *Body*
  Enthält die eigentliche Nachricht.

Abbildung 3 zeigt wie ein entfernter Methodenaufruf mit SOAP realisierbar ist. Im Body-Block steht der eigentliche Methodenaufruf. Es wird die Methode

```
withdraw(int  account, double amount)
```

beim Server aufgerufen.
Im Header-Block könnte z.B. eine digitale Signatur stehen, um den Aufrufer zu authentifizieren und um sicherzustellen, dass die Nachricht nicht verändert wurde.

2.2. WSDL

WSDL ist die Schnittstellenbeschreibung für Web-Services. WSDL ist vergleichbar mit IDL in CORBA. Mit WSDL wird beschrieben, wo sich der Dienst befindet, wie der Dienst heißt, und welches Übertragungsprotokoll verwendet wird. Außerdem sind sowohl die Methoden mit Parametern und Rückgabewerten aufgelistet, als auch die verwendeten Nachrichten und Datentypen definiert.
WSDL ist ein XML-Dokument mit folgenden Elementen ([Hau02]):

- *Types*
  Definition der verwendeten Datentypen.
- *Messages*
  Auflistung der Nachrichten mit ihren Inhalten, also den verwendeten Datentypen.
- *Operations*
  Zusammenfassung der Nachrichten zu Operationen. Ein Methodenaufruf besteht z.B. aus ein Input- und Output-Nachricht.
- *Port Types*
  Definition der Schnittstellen.
- *Binding*
  Bindung einer Schnittstelle an das Übertragungsprotokoll.
- *Port*
  Schnittstelle einer konkreten Web-Service-Instanz mit Bindung und Adresse.
- Service
  Beschreibung einer konkreten Dienstinstanz mit Namen und verwendeten Ports.

Abbildung 4 zeigt den hierarchischen Aufbau eines WSDL-Dokumentes:

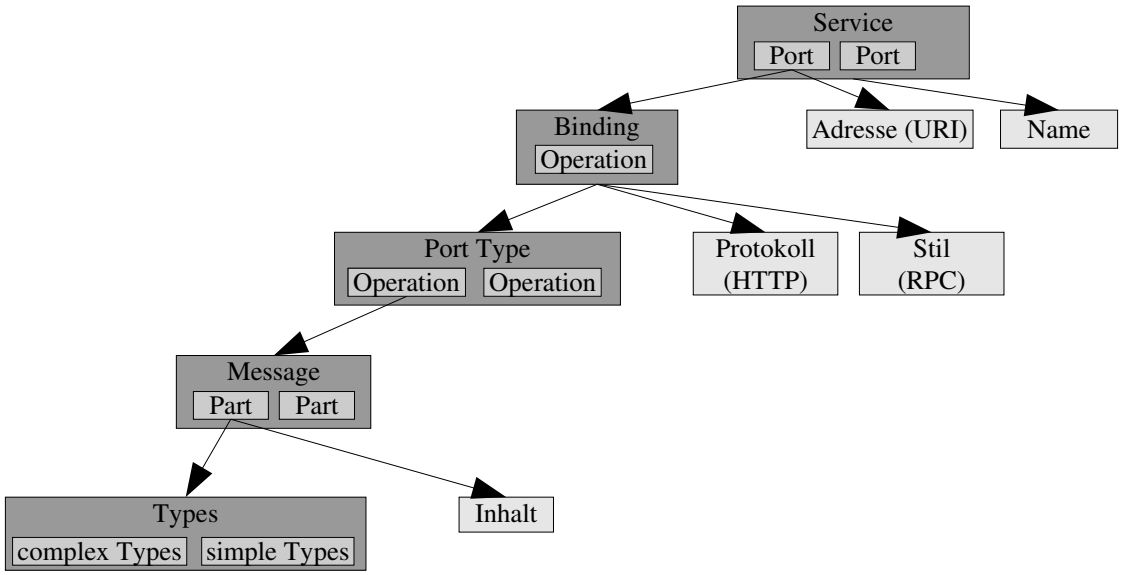

Abbildung 4: hierarchischer Aufbau eines WSDL-Dokumentes

2.3. UDDI

Die UDDI Spezifikation ([BCR03]) definiert einen Registry-Dienst für Web-Services. Der UDDI-Registry-Dienst ist ein Web-Service zum Beschreiben, Suchen und Publizieren von Web-Services und Geschäftseinheiten.
Ein UDDI-Registry-Dienst bietet folgende Dienstleistungen an:

- *White Pages*
  Namensregister
- *Yellow Pages*
  Kategorienregister
- *Green Pages*
  liefern technische Details zum Dienstzugang

White Pages sind einfache Nameserver wie etwa DNS. Sie liefern Angaben zu Unternehmen und Services. Yellow Pages können genutzt werden um nach Services oder Unternehmen einer bestimmter Klassifikation zu suchen. Eine Anfrage könnte z.B. lauten: finde alle Kinos in Ulm. Green Pages werden genutzt, um technische Details (WSDL-Beschreibung) zum Dienstzugang zu erhalten. Mit Hilfe der WSDL-Beschreibung kann ein Klient den Dienst in Anspruch nehmen.

3. Web-Services mit Java

3.1. Java Web Services Developer Pack

Java WSDP ist ein Toolkit zum Entwickeln von Web-Services und enthält folgende Pakete ([JWS04]):

- SOAP with Attachments API for Java (SAAJ)
  Erstellung von SOAP Nachrichten mit Anhängen.
- Java API for XML based RPC (JAXRPC)
  Entfernte Methodenaufrufe über XML-Nachrichten (SOAP). Es ist ein Compiler enthalten für die Java ⇔ WSDL Konvertierung. Außerdem besteht die Möglichkeit zur Integration mit EJB, Servlets und JSP.
- XML and Web Services Security
- Und vieles mehr.

3.2. Apache Axis

Axis ist ein SOAP-Engine von Apache für die Ausführung von Web-Services ([Axi04]). Mitgeliefert werden zwei Compiler für die WSDL ⇔ Java Konvertierung: Java2WSDL ist ein Compiler für die Generierung des WSDL-Dokuments anhand eines Java-Interface, WSDL2Java ist ein Compiler für die Generierung des Klienten-Stubs und des Service-Wrappers. Axis beinhaltet auch ein Admin-Client-Tool für das Einspielen von Web-Services. Die Entwicklung entfernter Objekte ist ähnlich wie bei RMI: statt rmic wird WSDL2Java verwendet, um die nötigen Klassen zu generieren.
Apache Axis ist die Implementierung des JAXRPC API und kann in einen Application Server (z.B. Tomcat) integriert werden. Der Vorteil bei der Verwendung von Apache-Axis

ist, dass der Java-Entwickler sich nicht um SOAP und WSDL kümmern muss, er kann sich wie gewohnt auf die Implementierung des Dienstes und des Klienten konzentrieren. Apache-Axis stellt außerdem noch eine einfache Möglichkeit dar, vorhandene Applikationen auf Web-Services umzustellen.

4. Zusammenfassung

Web-Services ermöglichen einen standardisierten Zugriff auf Applikationen über das Internet. Es ist eine Art „maschinenlesbares“ World Wide Web. Dadurch erreicht man eine Kommunikation zwischen den Applikationen unabhängig von Sprachen, Plattformen und Protokollen. Aus einfachen Web-Services lassen sich komplexere zusammenbauen, erleichtert vor allem durch das dynamische Auffinden von geeigneten Web-Services.

Vorteile von Web-Services:

- Akzeptierte Technologie zur Interaktion in heterogenen Systemen.
- Breite Unterstützung durch die Industrie.
- Geringe Integrationskosten vorhandener Systeme

Nachteile von Web-Services:

- Kommunikationsoverhead
- Nicht ausgereifte Standards
- Komplexität und Anzahl der beteiligten XML-Dokumente
- Eng gekoppelte Systeme schwer realisierbar

Literaturverzeichnis

[Hau02]: F. Hauck, Moderne Konzepte Verteilter Systeme, Universität Ulm, 2002
http://www-vs.informatik.uni-ulm.de/teach/ws02/mkvs/2002w-MKVS-Script-D.pdf [Juni, 2004]

[Smi04]: A. Schmietendorf, Web Services, Otto-von-Guericke-Universität Magdeburg, 2004
http://ivs.cs.uni-magdeburg.de/~schmiete/lehre/vorlesung/paper/web_services_ws03_1.pdf
[Juni, 2004]

[BCR03]: T. Bellwood, L. Clément, C. Riegen, UDDI Version 3.0.1, OASIS, 2003
http://uddi.org/pubs/uddi-v3.0.1-20031014.pdf [Juni, 2004]

[JWS04]: Java Web Services Developer Pack, Sun, 2004
http://java.sun.com/webservices/jwsdp/index.jsp [Juni, 2004]

[Axi04]: Axis, Apache, 2004
http://ws.apache.org/axis/index.html [Juni, 2004]