

# Web-Dienste

## Block Web-Dienste

### heutige Vorlesung

- Was sind Web-Dienste (Web Services)?
- diensteorientierte Architekturen
- Was ist SOAP, WSDL und UDDI?
- Entfernte Prozeduraufrufe (RPCs) vs. Messaging

### 16.6.

- Nachrichtenformat SOAP im Detail

### 23.6.

- Schnittstellenbeschreibung WSDL im Detail

### 30.6.

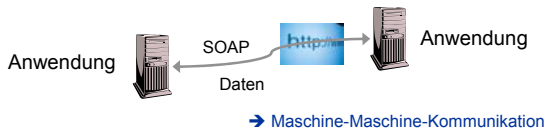
- Web-Dienste in der Praxis

## Was sind Web-Dienste (Web Services)?

### traditionelle Web-Anwendung



### Web-Dienst (Web Service)



## Beispiel: Google ohne Browser



With Google Web APIs, your computer can do the searching for you.

- Google auch als Web-Dienst: Suche und Rechtschreibkorrektur.
- Anwendung → Google: Suchanfrage als SOAP-Nachricht.
- Google → Anwendungsprogramm: Suchergebnis als SOAP-Nachricht.

## Beispiel: Google ohne Browser

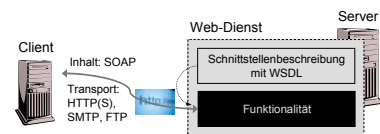
Google kann also aus Anwendungsprogramm heraus aufgerufen werden, um z.B.:

- in periodischen Abständen zu bestimmten Thema nach neuen Webseiten zu suchen
- automatisch neue Trends im WWW zu identifizieren
- die Rechtschreibkorrektur von Google zu nutzen



→ <http://www.google.com/apis/>

## Definition



Ein **Web-Dienst (Web Service)** ist eine Software, die

1. auf einem Server bereitgestellt wird,
2. eine bestimmte Funktionalität als Blackbox zur Verfügung stellt,
3. über gängige Internet-Protokolle unter Benutzung von SOAP zugreifbar ist und
4. über eine mit WSDL beschriebene Schnittstelle verfügt.

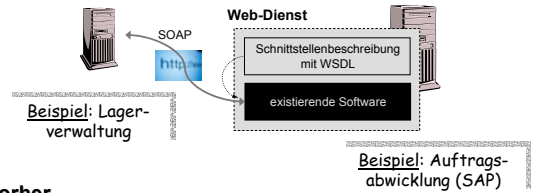
## Eigenschaften von Web-Diensten

- implementieren *keine neuen* Systeme
- Fassade für bestehende Systeme, um diese einfach zuzugreifen
- nutzen gängige Internet-Protokolle wie HTTP(S), SMTP und FTP
- verwenden XML-Standards SOAP und WSDL
- unabhängig von Programmiersprachen und Betriebssystemen
- zwei Erscheinungsformen: entfernte Prozeduraufrufe (synchron) oder Messaging (asynchron)

© Klaus Schild, 2004

7

## Einfacher Zugriff auf Systeme



### vorher

- SAP-System von außen nicht zugreifbar
- Aufträge per Fax oder E-Mail

### als Web-Dienst

- SAP-System von außen über SOAP/HTTP zugreifbar

© Klaus Schild, 2004

8

## Alter Wein in neuen Schläuchen?

- Web-Dienste *keine* revolutionär neue Technologie
- große Ähnlichkeiten mit Corba

### Neu jedoch:

- alle bedeutenden IT-Unternehmen auf Standards geeinigt: SOAP/WSDL
- statt proprietäre Protokolle (wie IIOB und DCOM) gängige Internet-Protokolle (wie HTTP und SMTP)
- nicht nur RPCs, sondern auch Messaging

© Klaus Schild, 2004

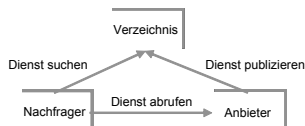
9

## Dienstorientierte Architektur

© Klaus Schild, 2004

10

## Dienstorientierte Architektur



- **publizieren (publish)**: Beschreibung eines Dienstes in einem Verzeichnis (*registry*) veröffentlichen.
- **suchen (find)**: Beschreibung eines Dienstes suchen.
- **abrufen (bind)**: Beschreibung des Dienstes verwenden, um Dienst abzurufen.

© Klaus Schild, 2004

11

## Öffentliches Verzeichnis

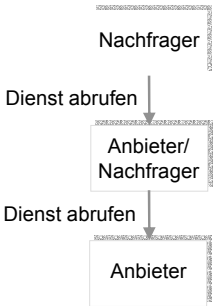


- Öffentliches Verzeichnis *nicht* zwingend notwendig;
- Auch *ohne* zentrales Verzeichnis kann Dienst gefunden werden.
- Schnittstelle kann z.B. auf Webseite des Anbieters veröffentlicht werden.
- Verzeichnisse heute kaum genutzt

© Klaus Schild, 2004

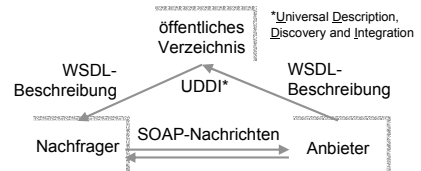
12

## Nachfrager vs. Anbieter von Diensten



- **Dienst-Anbieter** (*service provider*) bietet Dienste an.
- **Dienst-Nachfrager** (*service requestor*) nutzt Dienste anderer Anbieter.
- Anbieter von Diensten kann *gleichzeitig* Dienste anderer nutzen (und so Nachfrager) sein.
- Diese Begriffe sind deshalb relativ.

## Standards



- SOAP und WSDL allgemein akzeptiert
- allerdings umstritten, ob WSDL-Beschreibungen ausreichen
- UDDI umstritten und wenig genutzt



## Das Nachrichtenformat SOAP

HTML	SOAP
<html>	<Envelope>
<head>	<Header>
Zusatzinformationen	Zusatzinformationen
</head>	</Header>
<body>	<Body>
Inhalt: Webseite	Inhalt: XML-Daten
</body>	</Body>
</html>	</Envelope>

- SOAP-Nachricht = spezielles XML-Dokument
- Nachrichtinhalt (Body) und Zusatzinformation (Header) voneinander getrennt
- SOAP-Nachricht enthält Daten und keine Webseiten.

## SOAP-Nachricht



- **Body:** Nachrichtinhalt
- **Header:** optionaler Briefkopf, enthält Zusatzinformationen
- **Envelope:** umschließt Nachrichtinhalt (Body) und den Briefkopf (Header)
  - W3C-Standard
  - Seit SOAP 1.2 steht SOAP *nicht* mehr für Simple Object Access Protocol!

## Eine SOAP-Nachricht an Google

```

<Envelope xmlns="http://www.w3.org/2003/05/soap-envelope" ... >
  <Body>
    <doGoogleSearch xmlns="urn:GoogleSearch">
      <key xsi:type="xsd:string">43890489689</key>
      <q xsi:type="xsd:string">Eine Anfrage</q>
      <start xsi:type="xsd:int">0</start>
      <maxResults xsi:type="xsd:int">10</maxResults>
      ...
    </doGoogleSearch>
  </Body>
</Envelope>
  
```

- Google-Suche wird aufgerufen.
- kein Briefkopf
- Datentypen stammen von XML-Schema

## Übertragung von SOAP-Nachrichten

- meist mit HTTP oder HTTPS
- Request-Response-Verhalten von HTTP unterstützt entfernte Prozeduraufrufe (RPCs).
- mit HTTP auch RPCs über eine Firewall hinweg
- Übertragung aber auch z.B. mit SMTP möglich (*Messaging*).

# WSDL

## Web Services Description Language

### Servicebeschreibung

#### abstrakte Schnittstelle



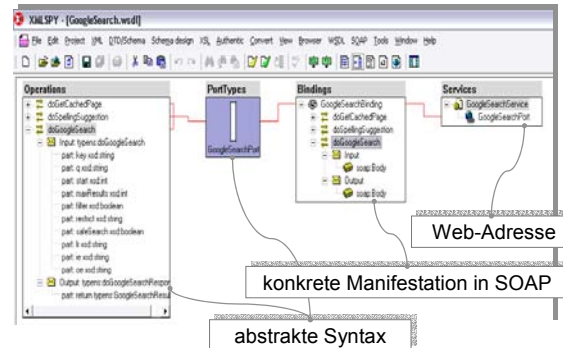
#### konkrete Schnittstelle



#### Web-Adressen von konkreten Operationen

- beschreibt Syntax einer Schnittstelle (→ IDL)
- WSDL-Beschreibung = spezielles XML-Dokument
- **Abstrakte Schnittstelle (port types)**: Syntax der Schnittstelle unabhängig von Nachrichtenformaten
- Syntax wird mit XML-Schema beschrieben.
- **konkrete Schnittstelle (binding)**: Abbildung auf unterstützte Nachrichtenformate

## Die WSDL-Beschreibung von Google



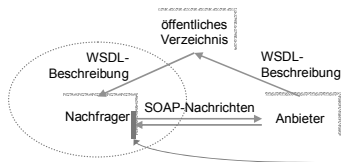
## Eigenschaften von WSDL

- beschreibt Schnittstelle eines Web-Dienstes und wo dieser abgerufen werden kann
- baut auf XML-Schema auf
- Syntax der Schnittstelle kann bis ins kleinste Detail festgelegt werden.
- Grundlegende Interaktionsmuster (wie Anfrage-Antwort) können beschrieben werden.

## Eigenschaften von WSDL

- Semantische Aspekte entweder gar nicht oder nur indirekt darstellbar.
- Verfügbarkeit: *nicht* darstellbar
- synchron vs. asynchron: nur *indirekt* über Protokoll-Binding (z.B. HTTP vs. SMTP) darstellbar

## Wie wird WSDL verwendet?



- aus WSDL-Beschreibung *automatisch* Schnittstelle zur Anwendung (Stubs) generieren
- z.B. mit .NET von Microsoft möglich
- Stubs abstrahieren von SOAP und vom Übertragungsprotokoll.
- Web-Dienst erscheint als *lokale* Bibliothek.

## Kommunikationsarten

## Verteilte Systeme



Ein **verteiltes System** ist eine Ansammlung von unabhängigen Computern, die den Nutzern als ein einziges kohärentes System erscheinen [Tanenbaum & van Steen 2002].

## Web-Dienste & verteilte Systeme



- Dienstorientierte Architektur ist spezielle Form eines verteilten Systems:
  - es werden SOAP-Nachrichten über gängige Internet-Protokolle ausgetauscht
  - Schnittstellen mit WSDL beschrieben
  - mit automatisch generierten Stubs kann von SOAP und verwendeten Protokoll abstrahiert werden

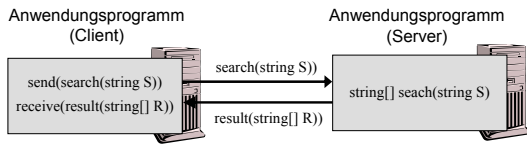
## Kommunikation in verteilten Systemen

drei verschiedene Arten der Kommunikation in verteilten Systemen:

- entfernte Prozeduraufrufe (RPCs)
  - Austausch von Nachrichten (*Messaging*)
  - Streams
- } von SOAP/WSDL unterstützt

## Entfernte Prozeduraufrufe

## Traditionelle verteilte Systeme

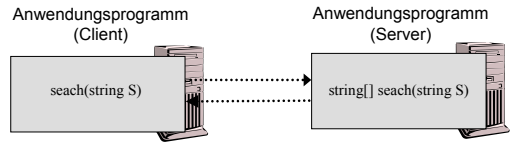


- traditionell basierte Kommunikation in verteilten Systemen auf Austausch von Nachrichten.
- Anwendungsprogramm ruft send- und receive-Operationen auf.
- Kommunikation im Anwendungsprogramm sichtbar.

© Klaus Schild, 2004

31

## Entfernte Prozeduraufrufe

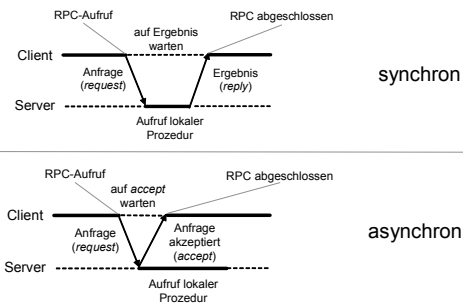


- Birrell & Nelson [84]: Prozedur kann auf entfernten Computer so aufgerufen werden, als ob sie auf dem lokalen Computer ausgeführt würde.
  - Ortstransparenz
  - Kommunikation für das Anwendungsprogramm *nicht* sichtbar.
- auch **Remote Procedure Call (RPC)** genannt

© Klaus Schild, 2004

32

## Synchrone vs. asynchrone RPCs



Unterscheidung auch für Web-Dienste relevant!

© Klaus Schild, 2004

33

# Messaging

## Messaging

- Anwendungen interagieren durch Austausch von Nachrichten miteinander
- Kommunikation in jeweiligen Anwendungen sichtbar
- verschiedene Formen des Messaging werden anhand folgender Kriterien unterschieden:
  1. Kommunikationsstruktur
  2. Interaktionsmuster
  3. verbindungsorientierte vs. persistente Kommunikation
  4. Synchronität
  5. Qualität (*quality of service*)
  6. Nachrichtenformat

Kriterien auch für Web-Dienste relevant!

© Klaus Schild, 2004

35

## Kommunikationsstruktur

- Anzahl und Organisation der Kommunikationspartner
- wichtigste Kommunikationsstrukturen:
  - Eins-zu-Eins-Kommunikation
  - One-to-Many-Kommunikation

© Klaus Schild, 2004

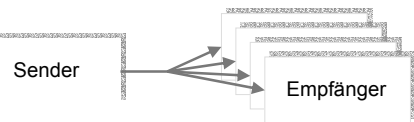
36

## Eins-zu-Eins-Kommunikation



- Sender sendet Nachricht an bestimmten Empfänger.
- Beispiel: Kunde sendet Bestellung per E-Mail an Firma

## One-to-Many-Kommunikation



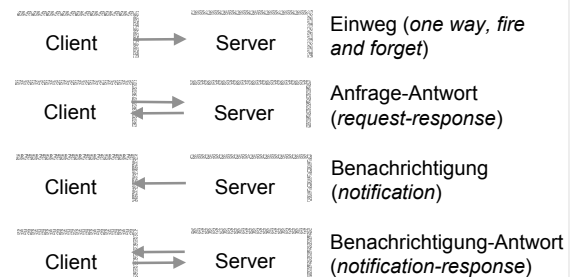
- Sender sendet identische Kopie gleichzeitig an mehrere Empfänger.
- Sender (*publisher*) veröffentlicht Nachricht zu einem bestimmten Thema (*topic*), zu dem sich die Empfänger (*subscriber*) angemeldet haben.
- auch *publish-subscribe* oder *topic-based messaging* genannt
- Beispiel: Mailing-Liste

## Messaging

Kriterien:

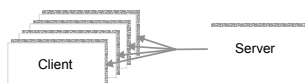
1. Kommunikationsstruktur
2. Interaktionsmuster
3. verbindungsorientierte vs. persistente Kommunikation
4. Synchronität
5. Qualität (*quality of service*)
6. Nachrichtenformat

## Grundlegende Interaktionsmuster



Beachte: „Antwort“ bezieht sich auf die jeweilige *Anwendung*, nicht auf das Netzwerk.

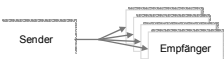
## Beispiel Börsenticker



- Interaktionsmuster: Benachrichtigung (*notification*)



- Kommunikationsstruktur: One-to-Many (*publish-subscribe*)



## Komplexe Interaktionsmuster

- Registrierung
- ← Bestätigung der Registrierung
- ← aktuelle Aktienkurse (Benachrichtigung)

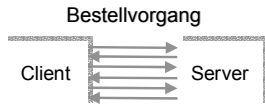


- Abfolge der Nachrichten wichtig
- z.B. keine Benachrichtigung ohne vorherige Registrierung
- auch als *Konversation* (*conversation*) bezeichnet

## Komplexe Interaktionsmuster



- Bestellanfrage mit spätestem Liefertermin
- ← Angebot mit zugesichertem Liefertermin
- Bestellung
- ← Bestätigung des Eingangs der Bestellung
- Bestätigung der Lieferung
- ← Rechnung



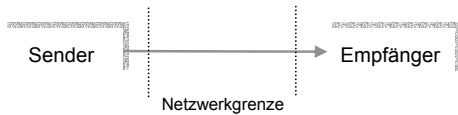
## Messaging



Kriterien:

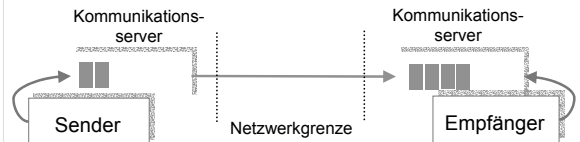
1. Kommunikationsstruktur
2. Interaktionsmuster
3. verbindungsorientierte vs. persistente Kommunikation
4. Synchronität
5. Qualität (*quality of service*)
6. Nachrichtenformat

## Verbindungsorientierte Kommunikation



- Sender und Empfänger kommunizieren direkt ohne Puffer miteinander.
- Sender und Empfänger müssen während der gesamten Übertragung präsent sein.
- verbindungsorientiert (*connection-oriented*) oder auch vergänglich (*transient*) genannt
- Beispiel: http

## Persistente Kommunikation



- Nachricht wird solange gespeichert, bis sie tatsächlich zugestellt wurde.
- *persistente* Kommunikation genannt
- Beispiel: E-Mail, MQSeries von IBM

## Messaging



Kriterien:

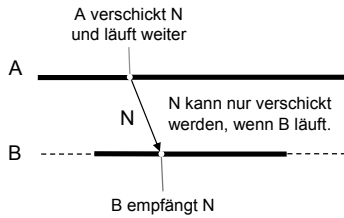
1. Kommunikationsstruktur
2. Interaktionsmuster
3. verbindungsorientierte vs. persistente Kommunikation
4. Synchronität
5. Qualität (*quality of service*)
6. Nachrichtenformat

## Synchronität



- auch beim Messaging Unterscheidung zwischen synchron oder asynchron
- persistente Kommunikation
  - typischerweise asynchron (wegen Puffer)
  - Beispiel: E-Mail
  - theoretisch aber auch synchron: Sender solange blockiert, bis er Bestätigung über *Empfang* der Nachricht bekommt
- verbindungsorientierte Kommunikation
  - auch in der Praxis sowohl synchron als auch asynchron

## asynchron + verbindungsorientiert

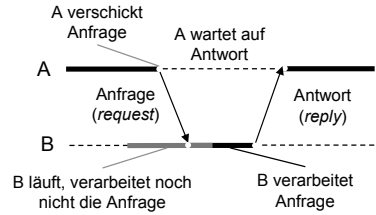


- Beispiel: UDP (User Datagram Protocol)

© Klaus Schild, 2004

49

## synchron + verbindungsorientiert I

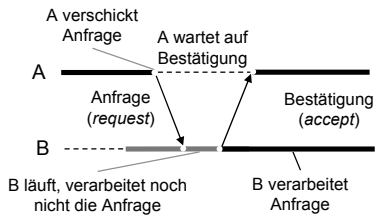


- auch **antwortorientiert** (*response-based*) genannt
- Beispiel: synchrone RPCs
- Implementierungsmöglichkeit für synchrone RPCs

© Klaus Schild, 2004

50

## synchron + verbindungsorientiert II



- auch **bestätigungsorientiert** (*delivery-based*) genannt
- Beispiel: *asynchrone* RPCs
- Implementierungsmöglichkeit für asynchrone RPCs

© Klaus Schild, 2004

51

## Messaging



### Kriterien:

1. Kommunikationsstruktur
2. Interaktionsmuster
3. verbindungsorientierte vs. persistente Kommunikation
4. Synchronität
5. Qualität (*quality of service*)
6. Nachrichtenformat

© Klaus Schild, 2004

52

## Qualität (*quality of service*)



### Kriterien

- Verfügbarkeit
- maximale Verzögerung (*latency*)
- maximale Last
- Toleranz gegenüber Ausfällen (z.B. ob Puffer in einer Datenbank gespeichert sind)
- Verschlüsselung
- Authentifizierung
- Nachrichten erreichen auf jeden Fall den Empfänger
- Nachrichten werden höchstens einmal zugestellt

© Klaus Schild, 2004

53

## RPC oder Messaging?



© Klaus Schild, 2004

54



### RPC

→ eng gekoppelte, starre Systeme

- + einfach, abstrahiert von Kommunikation
- nur Eins-zu-Eins-Kommunikation
- Client und Server müssen präsent sein.
- nicht skalierbar

### Messaging

→ lose gekoppelte, flexible Systeme

- abstrahiert nicht von Kommunikation
- + erlaubt auch One-to-Many-Kommunikation
- + Weder Sender noch Empfänger müssen präsent sein.
- + erlauben Priorisierung und Lastverteilung, dadurch skalierbar



### heutige Vorlesung

- ☑ Was sind Web-Dienste (Web Services)?
- ☑ dienstorientierte Architekturen
- ☑ Was ist SOAP, WSDL und UDDI?
- ☑ Entfernte Prozeduraufrufe (RPCs) vs. Messaging

### 16.6.

- Nachrichtenformat SOAP im Detail

### 23.6.

- Schnittstellenbeschreibung WSDL im Detail

### 30.6.

- Web-Dienste in der Praxis