



Forschungszentrum
Informatik



Universität
Karlsruhe (TH)



Information Process Engineering

Integration autonomer Systeme II: XML-basierter Datenaustausch und RDF

Andreas Schmidt

WS 2006/2007

- bisher: Informationsintegration aus einer individuellen Sicht
 - ▶ für **eine** spezielle (Portal-) Anwendungen werden Informationen aus Informationsquellen extrahiert
 - ▶ Asymmetrie: es existiert eine **zentrale** Stelle, von der aus alles koordiniert wird, die Datenmodelle, Schemata definiert und integriert
 - ▶ Wahrung der Autonomie der einzubindenden Quellen
- jetzt: allgemeinerer Ansatz
 - ▶ Symmetrie: **keine vorherrschende Interaktionsrichtung**
 - ▶ weiterhin: Autonomie der technischen Systeme
 - ▶ aber teilweiser **Verzicht auf organisatorische Autonomie**
 - man einigt sich auf „etwas“
 - ▶ meistens Verteilung des Implementierungsaufwandes
=> klare Spezifikationen erforderlich

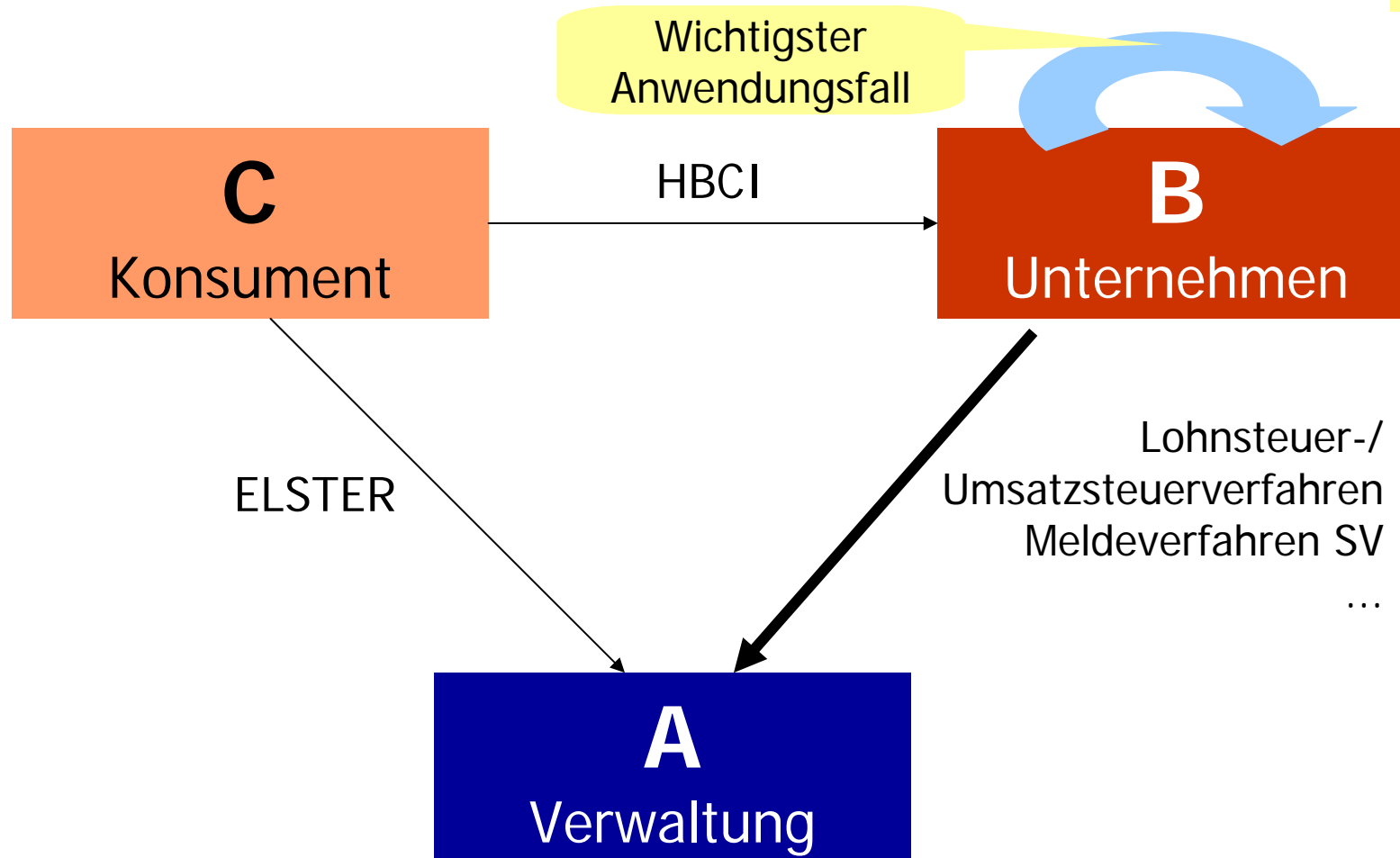
■ Datenaustausch

- ▶ Anforderungen und Probleme
- ▶ Traditionelle EDI-Lösungen
- ▶ XML-basierter Datenaustausch
- ▶ Validierung als wichtigstes (technisches) Teilproblem
 - XML Schema
 - Andere Validierungstechniken für XML

Datenaustausch



Beispiele nach Akteuren



- Grundmodell: Kooperation ist der Austausch
 - ▶ von Daten
 - ▶ zwischen einer Gruppe von Partnern
 - ▶ für einen bestimmten Kooperationszweck

- Austausch ist dabei geregelt
 - ▶ entweder zwischen einzelnen Partnern,
 - ▶ branchenweit
 - ▶ oder gar als internationaler Standard

= > **Spezifikation**

Spezifikation: Bestandteile



- Abgrenzung des **Einsatzbereiches**
 - Was ist Hintergrund und Zweck?
 - evtl. „Use Cases“
- **Informationsmodell**
 - Festlegung der Datenstrukturen
 - Welche Entitäten?
 - Wie werden die Entitäten beschrieben?
 - insbesondere: Identifikation von Entitäten
- **Prozessmodell**
- **Kommunikationsprotokoll**

Spezifikation: Informationsmodell



- **Informationsmodell** ist typischerweise Kern der Spezifikation
 - entspricht der Entwicklung des Domänenmodells bei der virtuellen Integration
 - sehr aufwendig und langwierig
 - letztlich Definition eines gemeinsamen Vokabulars

Spezifikation: Prozeßmodell

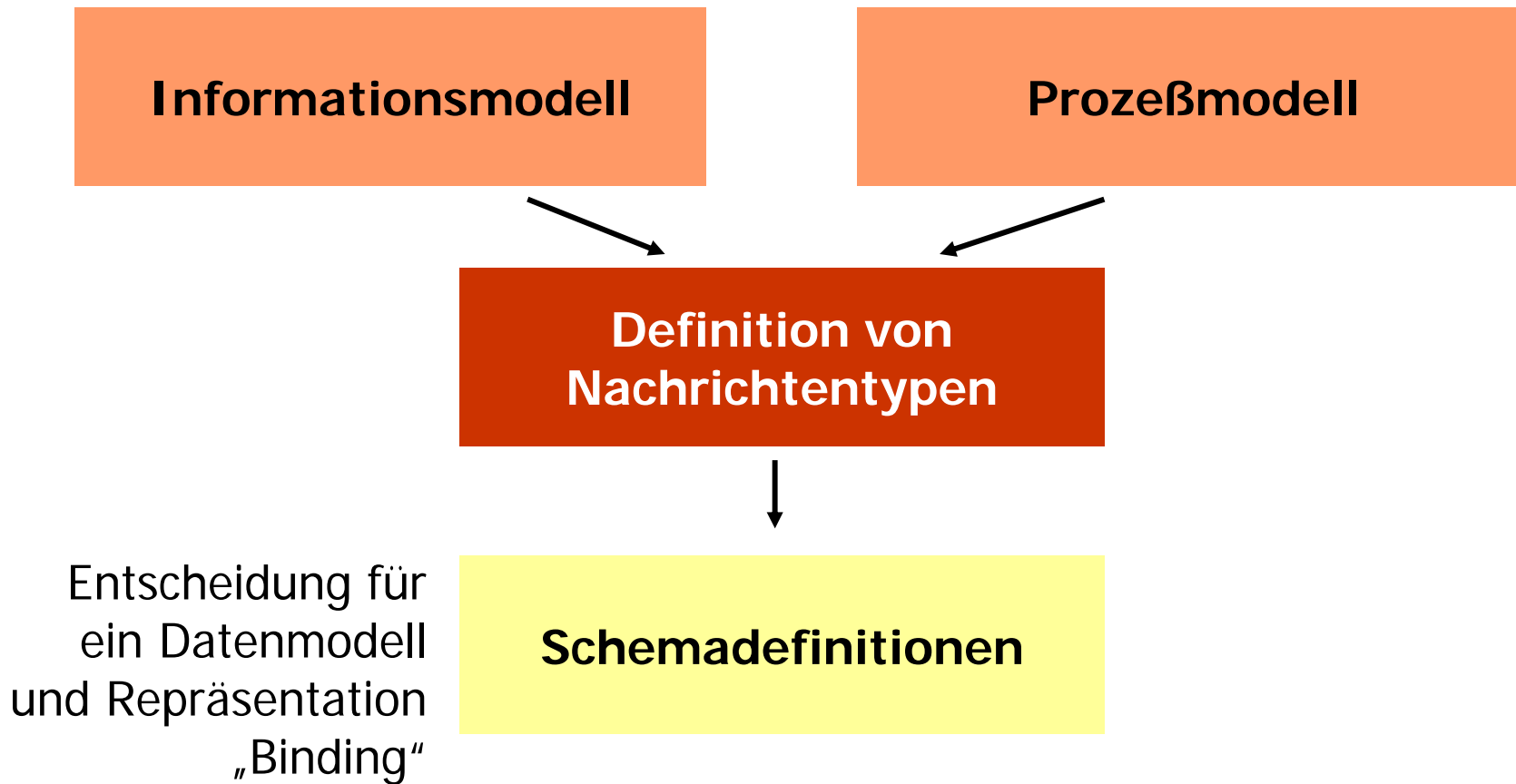


- **Prozeßmodell** hat zwei Aspekte
 - ▶ dahinterstehende Geschäftsprozesse
 - bildet den Hintergrund
 - umfassen insbesondere die internen Abläufe und die Schnittstellen nach außen
 - in diese muß das Austauschzenario integriert werden
 - Beispiel: typische Beschaffungs- und Lieferungsprozesse
 - ▶ abstraktes Austausch-„Protokoll“
 - das wird zwischen den Partnern spezifiziert
 - keine internen Prozesse

Spezifikation: Prozeßmodell (2)

- In jedem Fall wichtig:
 - ▶ Festlegung von Rollen
- Formale Prozeßmodellierung
 - ▶ mit zunehmender „Integrationstiefe“ und Komplexität anzuraten, obwohl derzeit noch kaum praktiziert
 - ▶ z.B. UML Interaktionsdiagramme, Zustandsdiagramme
 - ▶ andere Workflow-Modellierungstechnik vgl. Vorlesung WFMS

Datenaustauschformat



Spezifikation: Nachrichten

- Wie setzt man diese Prozesse um?
 - Repräsentation des Prozeßkontextes als semantische Zusatzinformation (welchen Zweck hat die Übermittlung)
 - Katalogdaten, Bestellungen, Rechnungen, ...

- **Nachrichtenbasierte Kommunikation**
 - Definition von Nachrichtentypen wie Bestellung, Reklamation
 - enthalten Informationsobjekte aus dem Informationsmodell als Parameter
 - Protokollautomat zur Abfolgeregelung
 - i.d.R. keine synchrone Verarbeitung (außer Quittungen)
 - separate Ergebnismnachrichten

- Vorstufe zur dienstbasierten Integration

Kommunikationsprotokoll

- Technische Perspektive
- Entwurfsentscheidungen
 - ▶ Austausch über Datenträger (z.B. Gesundheitskarte) vs. Netz
 - ▶ Zentrale Sammel- und Verteilstellen vs. Punkt-zu-Punkt
 - ▶ Transportprotokolle (FTAM, SMTP, FTP, HTTP, ...)
 - ▶ Sicherheit:
 - Authentifizierung (Gesundheitskarte, HealthProfessionalCard)
 - Verschlüsselung
 - Signatur
 - ▶ ...

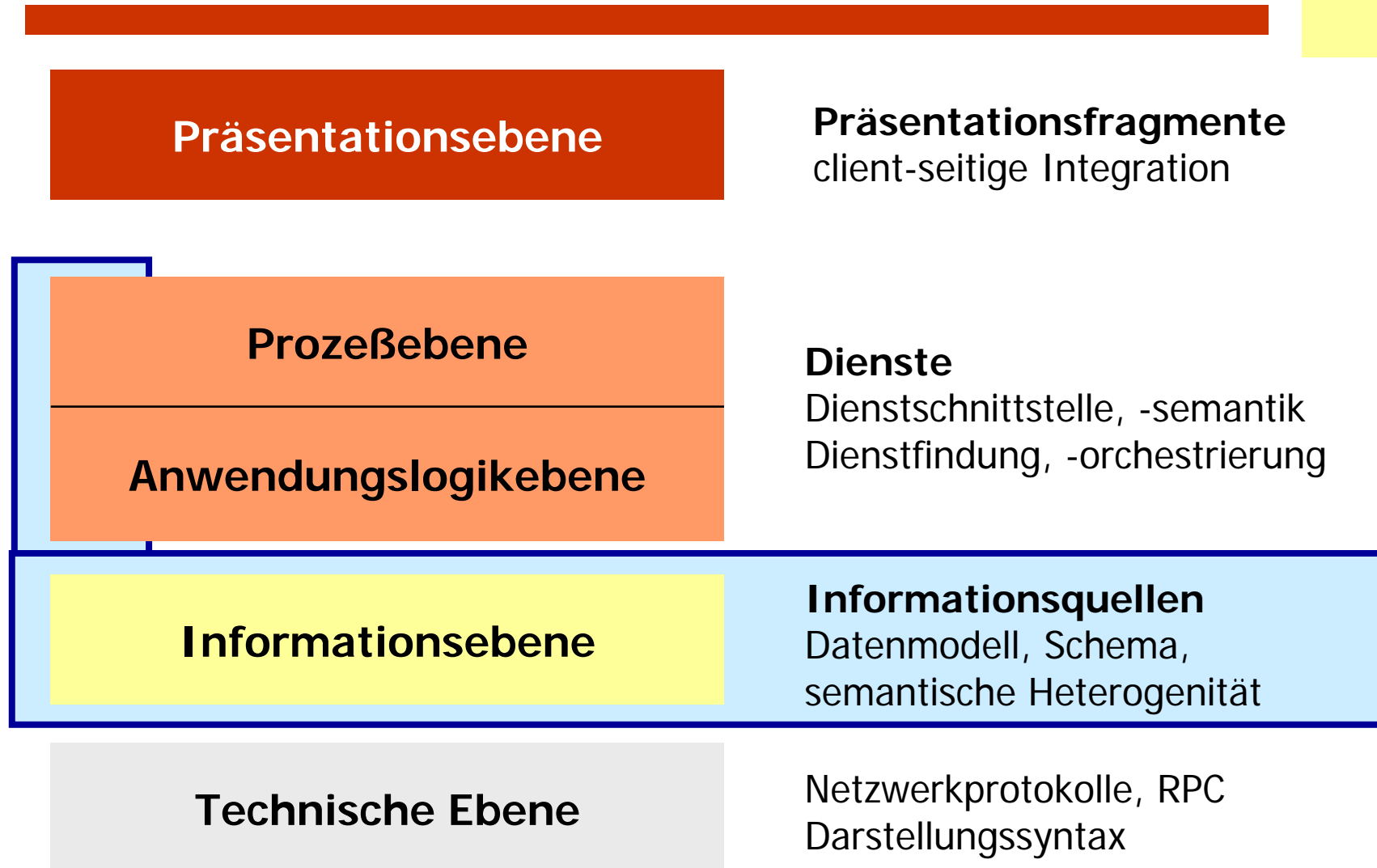
Was müssen die Partner tun?



- Informationsebene
 - ▶ Bereitstellung der Daten im gewünschten Schema
 - ▶ Verarbeitung der ausgetauschten Daten und Einbringung in das eigene System

- Anwendungs-/Prozeßebene (meist komplexer!)
 - ▶ Einbindung der Datenaustauschschritte in die eigenen Geschäftsprozesse
 - müssen nicht den der Spezifikation zugrundegelegten Prozessen entsprechen
 - ▶ Abstimmung der Anwendungslogik

Einordnung Datenaustausch in die Integrationsebenen



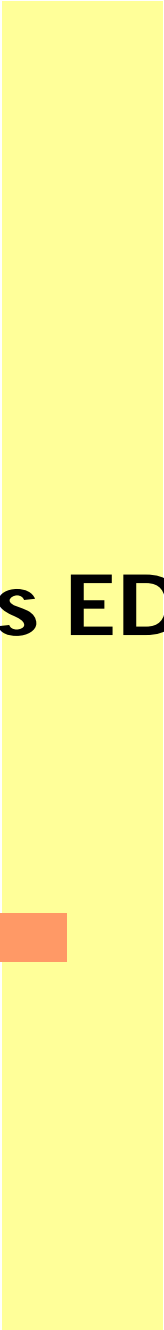
Datenaustausch vs. Informationsintegrationsansatz

- Generell: drei getrennte Welten
 - ▶ **Datenaustausch**: pragmatischer, aus der Praxis kommender Ansatz zum Austausch von Daten zwischen unabhängigen, aber kooperierenden Einrichtungen
=> Schnittstellen nach außen
 - ▶ **Data Warehousing (materialisierte Integration)**: unternehmensinterner Aufbau einer integrierten Datenbasis zur Verbesserung der Auswertung, zur Wissensgewinnung etc., dabei Entkopplung der Auswertung vom operativen Geschäft => Integration im Innern
 - ▶ **Virtuelle Informationsintegration**: Ansatz, der sich auf die Übersetzung von Anfragen und Ergebnissen zur Laufzeit kümmert
=> Integration von externen Informationsquellen

Datenaustausch vs. Informationsintegrationsansatz (2)

- Zusammenwachsen der unterschiedlichen Problembereiche und Lösungen
 - ▶ Enterprise Application Integration
 - ▶ Web-Service-basierte Lösungen
=> dienstorientierte Integration
- Deshalb Schwerpunkt dieser Vorlesung auf den Beitrag des Datenaustauschszenarios
 - ▶ Validierbarkeit der Spezifikationskonformität von Daten

Klassisches EDI



Austauschformate: EDI

■ Electronic Data Interchange

- ▶ einigermaßen standardisierter nachrichtenbasierter Datenaustausch
- ▶ unterteilt in unterschiedlichen Nachrichtengruppen, Nachrichtentypen, Segmente, Attribute
- ▶ Definition von speziellen Nachrichtentypen, z.B.
 - Angebot
 - Bestellung
 - Rechnung
 - Bezahlung

■ Unterschiedliche Ausprägungen

- ▶ ANSI X.12
- ▶ UN/EDIFACT
(**E**lectronic **D**ata **I**nterchange for **A**dministration, **C**ommerce and **T**ransport)

Beispiel: Rechnung

Absender:

Klick and Bau
Haid-und-Neu-Straße 10-14
76131 Karlsruhe

Empfänger:

Huber GmbH
Obstgasse 2
81549 München

Rechnungsnr. 200301 vom 08.01.2005

Ihre Bestellung O0010001 vom 23.12.2004

1 Bohrmaschine Bosch PSB 500 RE	43,06
Summe netto	43,06
MwSt 16%	6,89
Summe brutto	49,95

Beispiel: Rechnung (2)

UNA:+,?'

Syntaxinformationen

UNB+UNOA:2+KLICKANDBAU+HUBER+20030108:1001+200301081010'

UNH+INVOIC0001+INVOIC:D:93A:UN'

Sender und Empfänger (IDs), Datum

BGM+380+2003001+9'

Nachrichtentyp Rechnung

DTM+3+20030108+102'

Rechnung, Rechnungsnr., Original
Rechnungsdatum

RFF+ON+00010001'

Referenz auf Bestellnummer
Bestelldatum

DTM+4+20021223:102'

NAD+SE++Klick and Bau++Haid-und-Neu-Str.10-14+Karlsruhe++76131'

NAD+BY++Huber GmbH++Obstgasse 2+München++81549'

Adresse des Rechnungsempfängers

Adresse des Rechnungsstellers

Beispiel: Rechnung (3)

LIN+1++4711.001'

Rechnungsposition 1 mit Artikelnr. 4711.001

IMD+F+++::Bohrmaschine Bosch PSB 500 RE'

Beschreibung

QTY+47:1:PCE'

Berechnete Anzahl 1 (PCE=Stück)

MOA+43,06:750'

Nettopreis der Position

PRI+AAA:43,06'

Einzelpreis der Position

UNS+S'

Summensektion

MOA+79:43,06'

Netto-Gesamtsumme

MOA+124:6,89'

(Mehrwert-)steuer

MOA+128:49,95'

Brutto-Gesamtsumme

TAX+7+VAT+++::16+S'

Steuerart = MwSt, Satz 16%

UNT+18+INVOIC0001'

Abschluß der Rechnung, #Segmente=18

UNZ+1+200301081010'

Abschluß der Nachricht

- EDIFACT-Lösungen haben eine große praktische Bedeutung, vor allem bei
 - ▶ hohen Transaktionsvolumina
 - ▶ stabilen Beziehungen zwischen den Partnern

- Z.B.
 - ▶ Einsatz in der Automobilbranche (Odette)
 - ▶ Bankenwesen (SWIFT)
 - ▶ Kommunikation im deutschen Gesundheitswesen (§§301,302 SGB V)
 - Krankenkassen, Krankenhäuser, Arbeitgeber, Sonstige Leistungserbringer
 - mit massiven Einführungsproblemen

Probleme mit X12/EDIFACT



- Für dynamische Umgebungen schlecht geeignet
 - ▶ Zentralisierte Vergabe von Identifikationsnummern
 - ▶ Mangelnde Erweiterbarkeit der Nachrichten
- Fehlende standardisierte und maschinenverarbeitbare Nachrichtenbeschreibungssprache
 - ▶ Keine standardisierten und nachrichtentypunabhängigen Validierungswerkzeuge
 - ▶ Oft ungenaue Spezifikationen

Probleme mit X.12/Edifact (2)

- Schwer zu erlernen
 - ▶ deshalb häufige Probleme mit inkompatiblen Implementierungen
 - ▶ Und: teuer
- Branchenspezifische Lösungen
 - ▶ Keine universelle EDIFACT-Lösung (Odette, SWIFT) wie ursprünglich anvisiert
- Für kleinere Unternehmen oder Ad-Hoc-Geschäftsbeziehungen nicht geeignet

Datenaustausch mit XML



XML für den Datenaustausch



- Datenaustausch ist **der** Anwendungsfall für XML

- Warum?
 - ▶ XML universelle Beschreibungssprache
 - ▶ (fast) beliebiger Grad der Strukturierung von unstrukturiert bis stark strukturiert ausdrückbar
 - ▶ Internationalisierung (Unicode, wohldefinierte Kodierungen)
 - ▶ Verfügbarkeit generischer Werkzeuge
 - Generierung
 - Validierung
 - Transformation
 - Import
 - Präsentation
 - ▶ selbstbeschreibendes Dateiformat

Schema und Validierung in XML



- Dokumenttypdefinitionen (**DTDs**) als Erbe von SGML
- XML Schema als der neue Standard des W3C
- Alternative Vorschläge
 - ▶ RELAX NG
 - ▶ Schematron

XML Schema



- offizielle XML-Schemadefinitionssprache des W3C
- Seit Mai 2001 Empfehlung (recommendation)
 - ▶ XML Schema 1.1 aktuell in Arbeit
 - ▶ soll langfristig DTDs ganz ablösen
- Inkorporation der erweiterten Möglichkeiten (Typen, Schlüssel) in andere Standards wie XSL-T oder XPath
 - ▶ XPath 2.0, XSL-T 2.0 und XQuery: Working Draft
- mehr zu XML Schema im allgemeinen in der Vorlesung Datenbankeinsatz von Prof. Böhm

XML Schema: Validierung

■ Validierungskonzept ähnlich wie bei DTDs

- ▶ Schemadefinition weiterhin optional
- ▶ Verknüpfung mit einer Schemadefinition kann flexibel erfolgen
 - direkter Verweis im Instanzdokument an beliebiger Stelle

```
<kab:Kunde xmlns:kab= "http://www.klick-and-bau.com/schema/1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.klick-and-bau.com/schema/1.0
  http://www.klick-and-bau.com/schemas/schema_1_0.xsd">
  ...
</kab:Kunde>
```
 - Angabe bei der eigentlichen Validierung

XML Schema: Validierung mit Java XML Validator API

```
import javax.xml.transform.Source;
import javax.xml.transform.stream.StreamSource;
import javax.xml.validation.*;
import org.xml.sax.SAXException;

public class ValidateKlickandBau {

    public static void main(String[] args) throws SAXException, IOException {
        SchemaFactory factory =
            SchemaFactory.newInstance("http://www.w3.org/2001/XMLSchema");

        java.io.File schemaLocation = new java.io.File(„schema_1_0.xsd“);
        Schema schema = factory.newSchema(schemaLocation);
        Validator validator = schema.newValidator();

        Source source = new StreamSource(args[0]);

        try {
            validator.validate(source);
            System.out.println(args[0] + " is valid.");
        }
        catch (SAXException ex) {
            System.out.println(args[0] + " is not valid because " + ex.getMessage()); }
    }
}
```

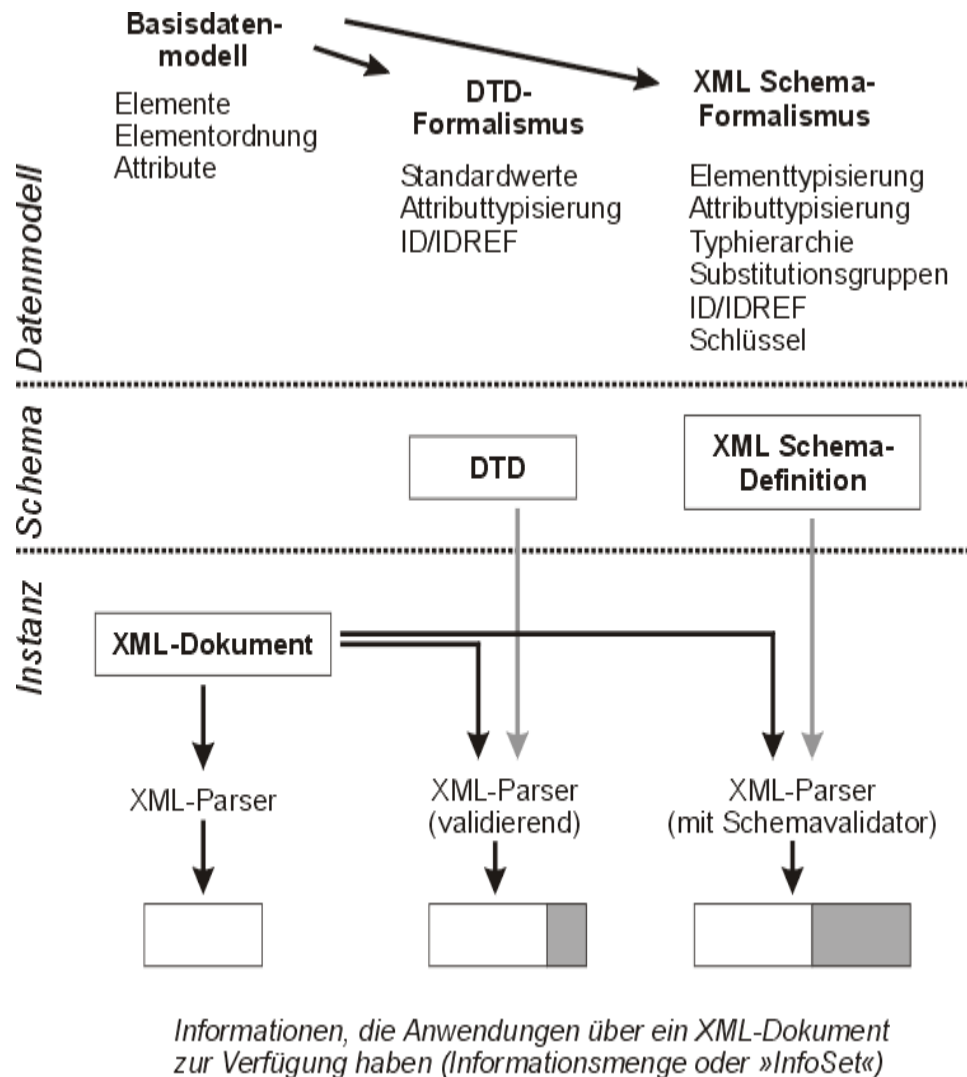
Ergebnis der Validierung



- XML Information Set als repräsentationsunabhängige Sicht auf ein XML Dokument
 - ▶ Validierung vergrößert die Informationsmenge, die Anwendungen über ein Dokument zur Verfügung steht
 - ▶ das Vorhandensein einer Schemadefinition entscheidet also über das Datenmodell!

- In der Validator-API von Java:
Zugriff auf ein „augmented document“

XML Schema: Auswirkung der Validierung



Validierung verändert das Datenmodell und die Informationsmenge, die über ein Dokument verfügbar ist.

XML Schema: Bausteine



Wiederverwendung
Modularisierung, Erweiterbarkeit

Konsistenzregeln
Schlüssel, Fremdschlüssel

Struktur
für komplexe Elemente

Datentypen
für Elementinhalte und Attribute

Konsistenzregeln



- Nullwerte
 - ▶ Unterschied zwischen `xsd:nil="true"` und leerem Element

- Eindeutigkeit (unique)

- Schlüsselbedingung (key)

- Fremdschlüsselbedingung (keyref)

Schlüsselkonzept

■ select

- ▶ definiert die Menge, für deren Elemente ein Schlüssel (bzw. eine Eindeutigkeitsbedingung) definiert werden soll (*scope*)

■ field

- ▶ definiert die Felder, die zum Schlüssel gehören
- ▶ nur die Kind-Achse von XPath ist erlaubt
 - keine relativen Schlüssel durch Hinzunahme »übergeordneter« Werte möglich

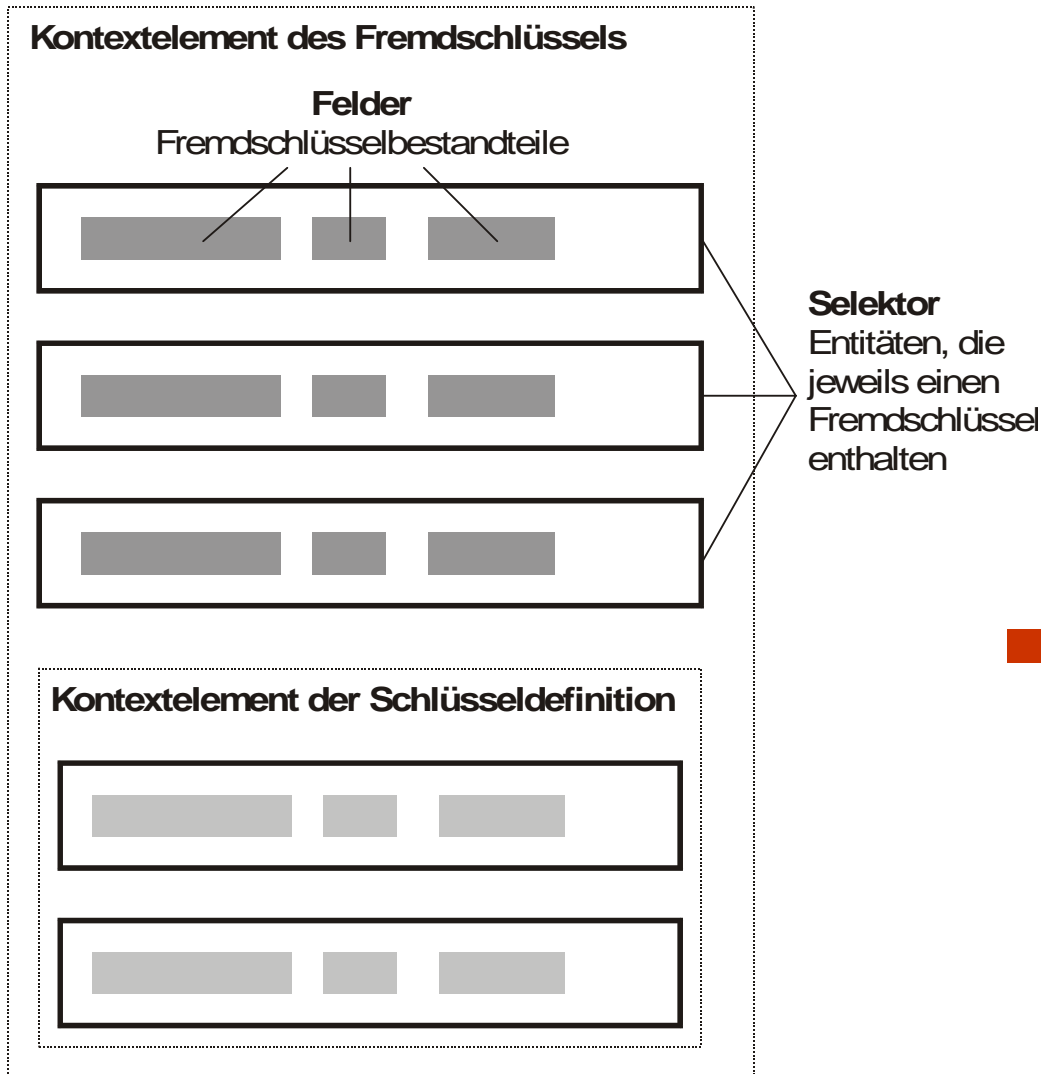
Schlüsselkonzept: Beispiel

```
<key name="A">  
  <selector xpath="//hersteller" />  
  <field xpath="name" />  
</key>
```

```
<keyref name="B" refer="A">  
  <selector xpath="//produkte" />  
  <field xpath="hersteller" />  
</keyref>
```

```
<herstellerliste>  
  <hersteller>  
    <name>Bosch</name>  
    ...  
  </hersteller>  
</herstellerliste>  
  
<produkte>  
  <produkt>  
    <name>...</name>  
    <hersteller>Bosch</hersteller>  
  </produkt>  
</produkte>
```

Schlüsselkonzept: Kontextelement



- Rel. Datenbanksicht
 - ▶ Kontext für Fremdschlüssel ist Datenbasis
 - ▶ Schlüsselkontext ist die Relation

Schlüssel und Datenaustausch

- Das Schlüsselkonzept von XML Schema ist nur eingeschränkt für die Datenaustauschproblematik zu gebrauchen:
 - ▶ Schlüssel sind auf das Dokument als Einheit beschränkt
 - ▶ Validierung von Problemen wie "Artikelnummer in der Bestellung muß einen im Katalog enthaltenen Artikel referenzieren"
 - kann nur bei versuchtem Einbringen in eine Datenbasis kontrolliert werden
 - Keine Validierungsmöglichkeit bei Ankunft der Bestellung
 - ▶ Vorschläge, hierfür XLink zu benutzen
 - Aber: kaum Werkzeuge, eher ein Exot

Bewertung von XML Schema



- sehr mächtiger Schemaformalismus für XML
 - ▶ fügt flexible Typisierung von Elementinhalten hinzu
 - ▶ erhöht Wiederverwendung
 - ▶ präzise Spezifikationen für Struktur und Datentypen

- aber:
 - ▶ recht komplexer Standard, insbesondere konkurrierende Ausdrucksmittel
 - ▶ Konsistenzregeln nicht für externe Daten zu gebrauchen
 - ▶ keine einfachen „Validierungsberichte“

Andere Schemaansätze für XML



Andere Schemaansätze



■ RELAX NG

- ▶ Entstanden aus TREX und RELAX
- ▶ Übernahme des Datentypenteils von XML-Schema
- ▶ Vereinfachung des Strukturteils
- ▶ Außerdem: kompakte Notation
- ▶ inzwischen ISO-Standard

■ Schematron

- ▶ basiert auf Zusicherungen
- ▶ kann Fehlerberichte einfach erzeugen
- ▶ seit Mai 2006 ISO-Standard
- ▶ versteht sich als orthogonal zu anderen Schemaansätzen

- Regelparadigma statt Grammatikorientierung
 - ▶ Strukturen werden nicht »erlaubt« durch das Angeben von Produktionsregeln einer Grammatik,
 - ▶ sondern beschränkt durch Validierungsregeln, die erfüllt sein müssen
- Grundelement von Schematron ist daher die Zusicherung (assertion)
 - ▶ als Argument ein XPath-Test
 - ▶ Verknüpfung mit Berichtstexten für Fehlerberichte
- Kann vollständig mittels XSL-T realisiert werden
 - ▶ einfach und kompakte Validierungssprache

Schematron: Beispiel

```
<schema>
  <pattern name="Produkte überprüfen">
    <rule context="produkt">
      <assert test="bezeichnung">Produktbezeichnung fehlt</assert>
      <assert test="//herstellerliste/hersteller[name=current()/hersteller]">
        Hersteller nicht in der Herstellerliste</assert>
      <assert test="floor(lagerbestand) = number(lagerbestand)">
        Anzahl nicht ganzzahlig</assert>
    </rule>
  </pattern>
</schema>
```

Schematron (2)

- Es lassen sich auch positive Prüfergebnisse ausgeben
 - ▶ report-Element

- Schematron will andere Schemaansätze nicht ersetzen, sondern ergänzen
 - ▶ Einbettung in <appinfo> von XML Schema
 - ▶ <xsd:annotation>
 - <xsd:appinfo>
 - <sch:pattern name="...">
 - <sch:rule context="d:Demo">
 - </xsd:appinfo>
 - </xsd:annotation>
 - ▶ Analoge Einbettung in RELAX NG Schemata

Schematron: Bewertung



- Wachsende Beliebtheit von Schematron
 - ▶ Zahlreiche Validatoren sind bereits verfügbar

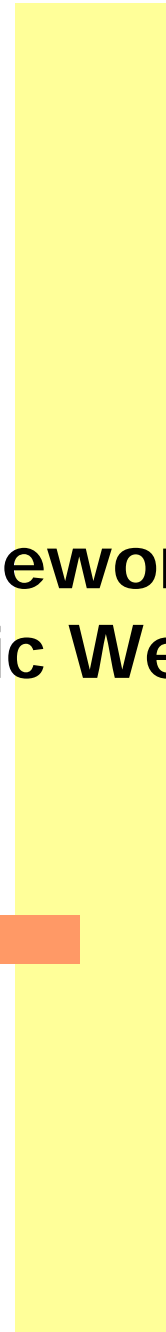
- Sehr einfach

- Hervorragend geeignet für Validierungsberichte

- Mögliche Perspektive
 - ▶ Formulierung der Grobstruktur mit XML Schema unter Verzicht zu starker Einschränkungen
 - ▶ Formulierung zusätzlicher Einschränkungen über Schematron

- Durch die javax.xml.validation API einheitliche Schnittstelle für unterschiedliche Validierungstechniken

Resource Description Framework als Datenmodell für das Semantic Web



Grenzen des XML-Datenmodells



- XML ist ein hervorragendes Datenmodell zur Repräsentation von semi- bis stark strukturierten Daten
 - ▶ bietet ausgereifte Validierungsmechanismen
 - ▶ Anfrage- und Transformationssprachen

- aber ...
 - ▶ XML bietet keine Mechanismen für die Verteilung von Daten
 - Aussagen über ein und dasselbe Produkt in verschiedenen Quellen
 - offene Welt
 - ▶ starke Fokussierung auf hierarchische Strukturen
 - Auswahl einer dominierenden Beziehung als Hierarchiebeziehung
 - Entwurfsentscheidung ohne wirkliche Semantik

Grenzen des XML-Datenmodells (2)

■ Beispiel:

```
<produkte>
  <produkt>
    <name>GBM 13-HRE</name>
    <hersteller>Bosch</hersteller>
  </produkt>
</produkte>
```

```
<hersteller>
  <name>Bosch</name>
  <produkte>
    <produkt>
      <produktname>
        GBM 13-HRE
      </produktname>
      ...
    </produkt>
  </produkte>
</hersteller>
```

unterscheidet sich die Semantik wirklich voneinander?

Grundprinzipien



- Ziel ist ein Datenmodell für das Web
 - ▶ möglichst nah an konzeptuellen Modellen
 - keine implementierungsspezifischen Entwurfsentscheidungen
 - Basis für semantische Beschreibungen
 - ▶ Grundphilosophie des Webs
 - offene Welt
 - verteilte und subjektive Beiträge
- Strikte Trennung von Objekten (=Ressourcen) und ihren Eigenschaften (=Aussagen/Statements über sie)

Resources und URIs

- Grundkonzept „Resources“:
 - ▶ beliebige physikalische oder abstrakte Objekte (Person, Buch, Web-Dokument ...)
 - ▶ identifiziert mittels Unified Resource Identifier (URI, RFC 2396)
- Literale für Werte (Zeichenkette, Zahl, Datum)
- URIs:
 - ▶ Wiederverwendung vorhandener URLs, E-Mail-Adressen etc.
`http://www.fzi.de/ipe, mailto:aschmidt@fzi.de`
 - ▶ URIs mit „Fragment Identifier“ (*Vokabular#Begriff*):
`http://www.fzi.de/glossar#Abteilungsleiter`

RDF Modell: Triples

■ Statement:

- ▶ „Andreas Schmidt arbeitet im Bereich IPE.“

■ Struktur:

- ▶ Subjekt (Resource)

`http://www.fzi.de/employees#ptomczyk`

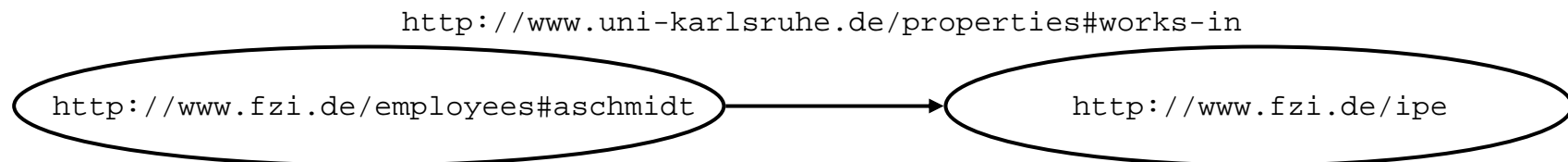
- ▶ Prädikat (Property \subset Resource)

`http://www.uni-karlsruhe.de/properties#works-in`

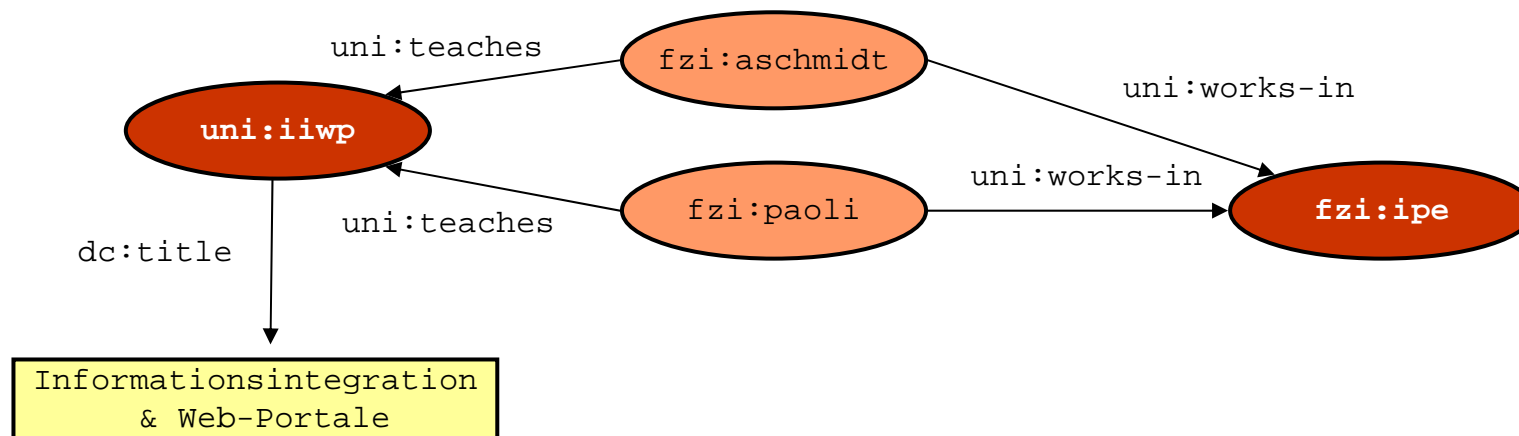
- ▶ Objekt (Resource | Literal)

`http://www.fzi.de/ipe`

■ Graph:



RDF: Graphstruktur



RDF: Formales Modell

■ Grundlegende Definitionen:

- ▶ Resources
- ▶ Properties \subset Resources
- ▶ Literals
- ▶ Statements = Properties \times Resources \times { Resources \cup Literals }

■ Deswegen spricht man auch vom Tripel-Modell

■ Einfache und klare formale Strukturen

- ▶ deswegen die Basis für automatische Schlussfolgerungen
- ▶ im Sinne der Logik nur zweistellige Prädikate ohne Funktionssymbole

- Das Tripel-Modell ist manchmal zu einfach
 - ▶ man möchte Beziehungen näher qualifizieren (z.B. works-in auf einen bestimmten Zeitraum einschränken)
 - ▶ man möchte Quellen von Aussagen angeben können (laut Eigenaussage ist das so)
- Man möchte Aussagen über Aussagen machen => hierzu müssen Statements auch als Ressourcen aufgefasst werden:

RDF und XML: RDF Syntax

- Um die Dinge kompliziert zu machen: XML ist eine mögliche Serialisierung eines RDF-Graphen:

```
<rdf:RDF xmlns=„...“ xmlns:rdf=„...“ xmlns:rdfs=„...“ xmlns:uni=„...“>

  <rdf:Description rdf:about=„#aschmidt“>
    <rdfs:label>Andreas Schmidt</rdfs:label>
    <uni:works-in rdf:resource=„#ipe“/>
    <uni:teaches rdf:resource=„http://www.uni-karlsruhe.de/iiwp“/>
  </rdf:Description>

  <rdf:Description rdf:about=„http://www.uni-karlsruhe.de/iiwp“>
    <rdfs:label>Informationsintegration und Web-Portale</rdfs:label>
  </rdf:Description>

</rdf:RDF>
```

- Ist damit RDF ein spezielles XML-Schema?

RDF und XML: RDF Syntax (2)

- Nein, denn das RDF-Datenmodell betrachtet folgende Serialisierung **semantisch als äquivalent** (im Gegensatz zu XML Schemata)

```
<rdf:RDF xmlns=„...“ xmlns:rdf=„...“ xmlns:rdfs=„...“ xmlns:uni=„...“>

  <rdf:Description rdf:about=„#aschmidt“>
    <rdfs:label>Andreas Schmidt</rdfs:label>
    <uni:works-in rdf:resource=„#ipe“/>
    <uni:teaches rdf:resource=„http://www.uni-karlsruhe.de/iiwp“/>
  </rdf:Description>
</rdf:RDF>

<rdf:RDF xmlns=„...“ xmlns:rdf=„...“ xmlns:rdfs=„...“ xmlns:uni=„...“>
  <rdf:Description rdf:about=„http://www.uni-karlsruhe.de/iiwp“>
    <rdfs:label>Informationsintegration und Web-Portale</rdfs:label>
  </rdf:Description>
</rdf:RDF>
```

- ... und deswegen ist RDF überhaupt erst interessant als Basis für das Semantic Web, das sich um syntaktische Probleme nicht kümmern will!

Alternative Repräsentationen

■ N-Triples

```
<http://www.fzi.de/employees#aschmidt> <rdfs:label> "Andreas Schmidt".  
<http://www.fzi.de/employees#aschmidt> <uni:teaches> <http://www.uni-karlsruhe.de/iiwp>.  
<http://www.uni-karlsruhe.de/iiwp> <rdfs:label> „Informationsintegration und Web-Portale“.
```

■ Turtle

```
@prefix uni: „http://www.uni-karlsruhe.de“.  
@prefix fzi: „http://www.fzi.de “.  
<fzi:employees#aschmidt>  
  rdfs:label „Andreas Schmidt“  
  uni:teaches  
  [  
    rdfs:label „Informationsintegration und Web-Portale  
  ]  
.
```

Anfragesprachen für RDF

■ SPARQL

- ▶ Konzept der Variablenbindungen mit konjunktiv verknüpften Tripelmustern

```
SELECT ?name ?vorlesung
WHERE  (?x <uni:teaches> ?y)
       (?y <rdfs:label> ?vorlesung)
       (?x <rdfs:label> ?name)
USING  uni FOR http://www.uni-karlsruhe.de
       rdfs FOR ...
```

■ RQL

- ▶ ebenso Variablenbindungen als Konzept
- ▶ erweiterte Möglichkeiten in Richtung Schemasprachen und Anfrage von Schemakonstrukten
- ▶ Pfadausdrücke zur Navigation auf den Graphenstrukturen



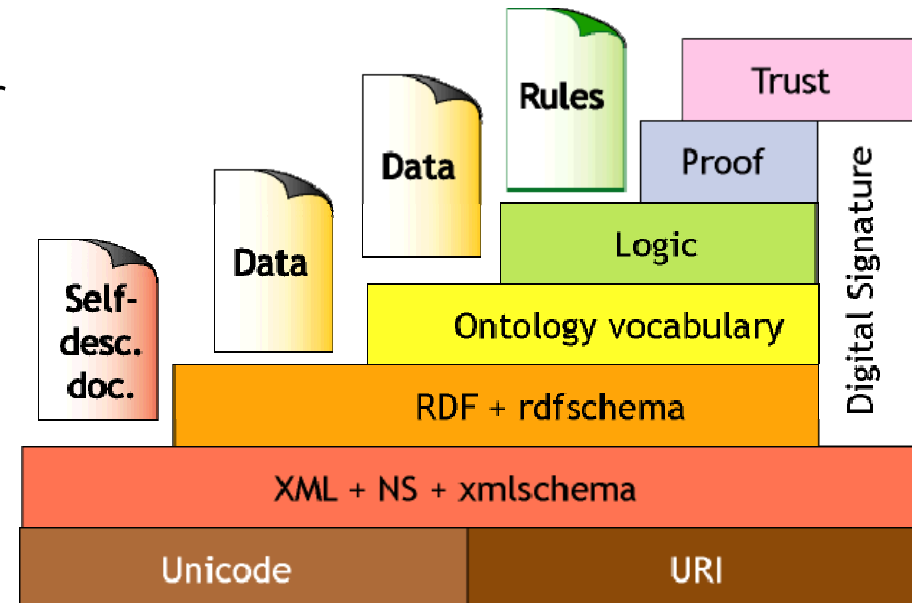
Fazit

Zusammenfassung

- XML hat sich im Bereich des Datenaustausches für neue Projekte als die Technologie der Wahl herauskristallisiert
 - standardisierte Spezifikationssprachen für Schemata
 - generische Werkzeuge (davon viele frei verfügbar) zu ihrer Verarbeitung
 - (leichte Integration/Migration in Web Service-Infrastrukturen)
- aber: XML ist nicht die Lösung aller Probleme
 - Verdrängt auf absehbare Zeit klassische EDI-Lösungen nicht
 - Die Verwendung von XML macht nicht die Systeme von sich aus interoperabel!
- RDF als Basis für bessere Repräsentation von Semantik

Weiteres Programm

- Ontologien als Werkzeug für die Informationsintegration
 - konzeptuelle Modelle
 - logische Schlußfolgerungen
 - Stichwort: **Semantic Web**



- Dienstorientierte Sicht auf die Integrationsproblematik
 - Dienstorientierte Integration mit **Web Services**

Frohe Weihnachten und ein Gutes Neues Jahr!

- Wassili Kazakos, Andreas Schmidt, Peter Tomczyk:
Datenbanken und XML, Springer, 2002
 - ▶ Kapitel 4
- RELAX-NG
 - ▶ <http://www.relaxng.org/>
- Schematron
 - ▶ <http://www.schematron.com>
- javax.xml.validation API
 - ▶ <http://www-128.ibm.com/developerworks/xml/library/x-javaxmlvalidapi.html>
- RDF Primer:
 - ▶ <http://www.w3.org/TR/rdf-primer/>