

7. XML

XML als Datenmodell

XML: Extensible Markup Language

Einführung in Datenbanksysteme

XML 7.1

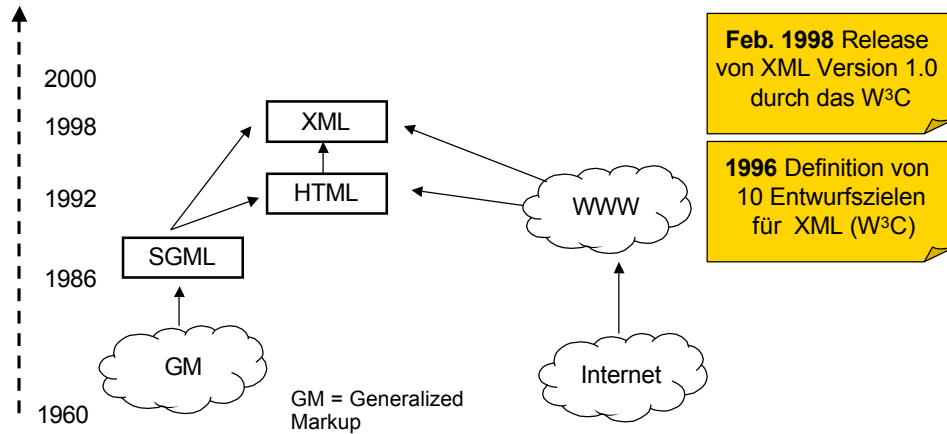
Motivation: XML zur Datenmodellierung

- ❑ XML: Textbasiertes Format für Daten (Austausch, Speicherung, Transformation etc.)
- ❑ Bestandteile eines XML-Dokuments:
 - Text, durchsetzt mit
 - Markups, d.h. Markierungen, die Teile des Textes hervorheben, z.B. um
 - den Text zu strukturieren,
 - Textteile für eine maschinelle Weiterverarbeitung zu markieren.
- ❑ XML-Dokumente (oder Teile davon) *können* ähnlich regulär strukturiert sein, wie Datensätze eines konventionellen Datenmodells, *müssen aber nicht* einem strengen vorgegebenen Schema folgen (daher: semi-strukturierte Daten).
- ❑ Speicherung von semi-strukturierten Daten in Datenbanken
 - Nutzung der Strukturdaten (Markup), um effizient auf Teile von Dokumenten (Text) zugreifen zu können

Einführung in Datenbanksysteme

XML 7.2

Historie von XML



The 10 Design Goals of XML

1. XML shall be straightforwardly usable over the Internet
2. XML shall support a wide variety of applications.
3. XML shall be compatible with SGML.
4. It shall be easy to write programs which process XML documents.
5. The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
6. XML documents should be human legible and reasonably clear.
7. The XML design should be prepared quickly.
8. The design of XML shall be formal and concise.
9. XML documents shall be easy to create.
10. Terseness in XML markup is of minimal importance.

Ausgangspunkt: unstrukturiertes Textdokument

Das Spektrum der sichtbaren Farben.
 In diesem Kapitel werden einige Eigenschaften des für den Menschen sichtbaren Farbspektrums vorgestellt. Die Farbe Rot z.B. befindet sich mit einer Wellenlänge zwischen 780-622 nm am unteren Ende des Spektrums. Die Farbe blau liegt am oberen Ende des sichtbaren Spektrums. Ihre Wellenlänge befindet sich im Bereich von 492-455 nm. Angaben zu anderen Farben des Spektrums finden Sie z.B. in Stöcker, Taschenbuch der Physik.

Eigenschaften:

- keine erkennbare Strukturierung des Textes
- Text nur für sachkundigen Leser verständlich
- maschinelle Bearbeitung (z.B. Layouterstellung für Handbuch) kaum möglich

XML-Dokument mit Markierung

```
<Kapitel><Titel>Das Spektrum der sichtbaren Farben</Titel>
In diesem Kapitel werden einige Eigenschaften des für den Menschen
sichtbaren Farbspektrums vorgestellt.
<Abschnitt>Die Farbe <Bezeichner>Rot</Bezeichner> z.B. befindet
sich mit einer Wellenlänge zwischen
<Zahlenwert>780-622 nm</Zahlenwert> am unteren Ende des
Spektrums.</Abschnitt>
<Abschnitt>Die Farbe <Bezeichner>blau</Bezeichner> liegt am oberen
Ende des sichtbaren Spektrums. Ihre Wellenlänge befindet sich im
Bereich von <Zahlenwert>492-455 nm</Zahlenwert>.</Abschnitt>
Angaben zu anderen Farben des Spektrums finden Sie z.B. in
<Referenz>Stöcker, Taschenbuch der Physik</Referenz>.</Kapitel>
```

Start-Tag
Content
End-Tag

Element

Eigenschaften:

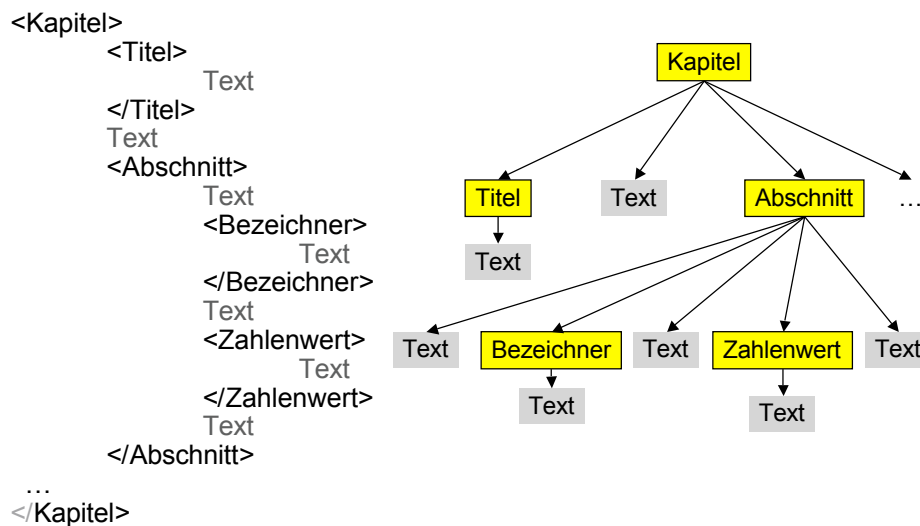
- Teile des Textes werden markiert
- Die Markierungen geben dem Leser Hinweise auf die Struktur des Textes
- Eine Maschine kann unterschiedliche Markierungen unterschiedlich behandeln

Nutzung der Markierung zur Strukturierung

```
<Kapitel>
  <Titel>
    Das Spektrum der sichtbaren Farben
  </Titel>
  In diesem Kapitel werden einige Eigenschaften des für den Menschen
  sichtbaren Farbspektrums vorgestellt.
  <Abschnitt>
    Die Farbe
    <Bezeichner>
      Rot
    </Bezeichner>
    z.B. befindet sich mit einer Wellenlänge zwischen
    <Zahlenwert>
      780-622 nm
    </Zahlenwert>
    am unteren Ende des Spektrums.
  </Abschnitt>
  ...
</Kapitel>
```

Elemente, die Teil des Contents anderer Elemente sind, sind hier stärker eingerückt.

Elementstruktur des XML-Dokuments



Visualisierung der Struktur

Kapitel	
Titel	Das Spektrum der sichtbaren Farben
Text	In diesem Kapitel werden einige Eigenschaften des für den Menschen sichtbaren Farbspektrums vorgestellt.
Abschnitt	
Text	Die Farbe
Bezeichner	rot
Text	z.B. befindet sich mit einer Wellenlänge zwischen
Zahlenwert	780-622 nm
Text	am unteren Ende des Spektrums.
Abschnitt	
Text	Die Farbe
Bezeichner	blau
Text	liegt am oberem Ende des sichtbaren Spektrums. Ihre Wellenlänge befindet sich im Bereich von
Zahlenwert	492-455 nm
Text	Angaben zu anderen Farben des Spektrums finden Sie z.B. in
Referenz	Stöcker, Taschenbuch der Physik
Text	-

Eigenschaften:

- XML-Element **Kapitel** enthält Text und einige andere XML-Elemente (**Titel**, **Abschnitt**, **Referenz**, **Bezeichner**, **Zahlenwert**)
- **Abschnitt** enthält Text und/oder weitere Elemente

Interpretation der Markierung

- Die Interpretation der Markierung ist nicht vorgegeben.
- Sie wird durch den Prozessor bestimmt, der die Markierung verarbeitet.
- Ein Prozessor zur Formatierung von Texten könnte z.B. mit folgender Interpretation (einem sogenannten Stylesheet, vgl. Folie 42ff) parametrisiert werden:

• <Titel>	□ größere Schrift
• <Abschnitt>	□ Leerzeile vorweg
• <Bezeichner>	□ Kursivschrift
• <Zahlenwert>	□ andere Schriftart
• <Referenz>	□ eckige Klammern

Das Spektrum der sichtbaren Farben

In diesem Kapitel werden einige Eigenschaften des für den Menschen sichtbaren Farbspektrums vorgestellt.

Die Farbe *Rot* z.B. befindet sich mit einer Wellenlänge zwischen 780-622 nm am unteren Ende des Spektrums.

Die Farbe *Blau* liegt am oberen Ende des sichtbaren Spektrums. Ihre Wellenlänge befindet sich im Bereich von 492-455 nm

Angaben zu anderen Farben des Spektrums finden Sie z.B. in [Stöcker, Taschenbuch der Physik].

Derselbe Text – unterschiedlich markiert

<Farben>Das Spektrum der sichtbaren Farben
 In diesem Kapitel werden einige Eigenschaften des für den Menschen sichtbaren Farbspektrums vorgestellt.
 <Rot>Die Farbe Rot z.B. befindet sich mit einer Wellenlänge zwischen <Wellenlänge>780-622 nm</Wellenlänge>am unteren Ende des Spektrums.</Rot>
 <Blau>Die Farbe blau liegt am oberen Ende des sichtbaren Spektrums. Ihre Wellenlänge befindet sich im Bereich von <Wellenlänge>492-455 nm</Wellenlänge></Blau>
 Angaben zu anderen Farben des Spektrums finden Sie z.B. in <Stöcker></Stöcker></Farben>

Eigenschaften:

- Art der Markierung hat sich geändert.

Interpretation der Markierung

<Farben>Das Spektrum der sichtbaren Farben
 In diesem Kapitel werden einige Eigenschaften des für den Menschen sichtbaren Farbspektrums vorgestellt.
 <Rot>Die Farbe Rot z.B. befindet sich mit einer Wellenlänge zwischen <Wellenlänge>780-622 nm</Wellenlänge>am unteren Ende des Spektrums.</Rot>
 <Blau>Die Farbe blau liegt am oberern Ende des sichtbaren Spektrums. Ihre Wellenlänge befindet sich im Bereich von <Wellenlänge>492-455 nm</Wellenlänge></Blau>
 Angaben zu anderen Farben des Spektrums finden Sie z.B. in <Stöcker></Stöcker></Farben>

Eigenschaften:

- Einige Markierungen werden als Farbe interpretiert.
- Andere werden nicht interpretiert oder können nur in einem anderen Kontext interpretiert werden (z.B. Wellenlänge oder Stöcker).

Bestandteile eines XML-Dokuments

❑ Text:

- Wird vom XML-Parser nicht weiter verarbeitet.
- Kann durch XML-Tags markiert werden.

❑ Elemente:

- Werden im Quelltext durch Tags (Elementname in spitzen Klammern), die den zu markierenden Dokumentteil klammern, gekennzeichnet, Beispiel: <Wert>95</Wert>.
- Werden zur Strukturierung des Dokumentes eingesetzt,
- Erscheinen in der Baumansicht als benannter Knoten,
- Können Text, weitere Elemente oder eine Kombination aus beiden enthalten,
- Werden von XML-Parsern erkannt und zum Aufbau von Baumstrukturen genutzt.

• Können Attribute besitzen, Beispiel:

```
<Projekt Nr="P200"> <Titel>ADAC Kundenstamm</Titel> </Projekt>
```

– Attributwerte beschreiben Eigenschaften von konkreten Elementen,

– Einige Attributtypen können von XML-Parsern zu Konsistenzprüfungen verwendet werden (IDs und IDREFs).

Attribut

Dokumentenzentrierte vs. datenzentrierte XML-Nutzung

❑ Dokumentenzentrierte XML-Nutzung – Geringe oder keine Regularität in der Verwendung von XML-Elementen:

- Komplexe Schachtelung von Elementen,
- Große "unmarkierte" Dokumentteile,
- nicht oder nur teilweise spezifizierter Dokumenttyp.

❑ Datenzentrierte XML-Nutzung – Hohe Regularität der Daten:

- XML-Dokument enthält Kollektion von gleichartigen Elementen.:
- Komplexität der "Datensätze" ist meistens beschränkt:
 - Geringe Schachtelungstiefe.
 - Schachtelungen gleichnamiger Elemente werden vermieden.
- Wenige oder keine optionalen Substrukturen.
- Textdaten nur innerhalb von Blattelementen.
- Dokumentstruktur ähnelt Datensätzen oder Kollektionen von Datensätzen einer konventionellen Datenbank

Beispiel: Dokumentenzentrierte XML-Nutzung

<Projekte>

Zur Zeit arbeitet unsere Abteilung an mehreren Projekten.

<Projekt Nr="P100">

Das Projekt <Titel>DB Fahrpläne</Titel> wurde am 1. Januar begonnen und verfügt über ein Budget von <Budget>300.000 €</Budget>. Falls zusätzliche Entwickler benötigt werden, können evtl. noch Entwickler aus anderen Projekten, wie z.B. aus dem Projekt

<Projekt Nr="P200"> <Titel>ADAC Kundenstamm</Titel> </Projekt>abgezogen werden.

</Projekt>

</Projekte>

- Projekt-Elemente besitzen ein Attribut "Nr".
- Beide Projekte besitzen ein Subelement "Titel".
- Ein Projekt verfügt über ein "Budget", das andere nicht.
- Das Projekt-Element enthält Text, der sich keinem der beiden Subelemente "Titel" oder "Budget" zuordnen läßt.
- Ein Projekt-Element darf weitere Projekt-Elemente enthalten.

Beispiel: Datenzentrierte XML-Nutzung

<Projekte>

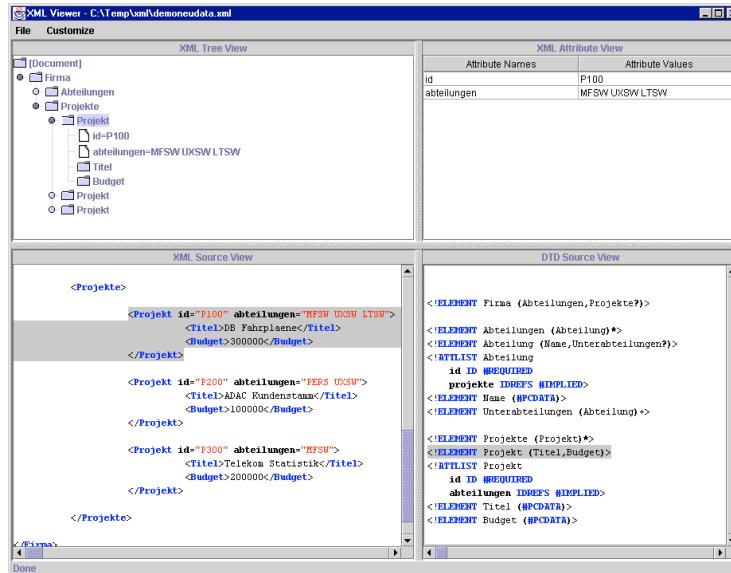
```
<Projekt Nr="P100">
  <Titel>DB Fahrpläne</Titel>
  <Budget>300000</Budget>
</Projekt>
```

```
<Projekt Nr="P200">
  <Titel>ADAC Kundenstamm</Titel>
</Projekt>
```

</Projekte>

- Projekt-Elemente besitzen ein Attribut "Nr".
- Projekt besitzt ein Subelement "Titel".
- Manche Projekte haben ein Subelement "Budget".
- "Projekte" ist eine Kollektion, die ausschließlich "Projekt"-Elemente enthält.
- Typischer Anwendungsfall: Export homogen strukturierter Datenbestände (z.B. Relationen) in ein Software-unabhängiges Datenformat

Präsentation: Baumansicht, Quelltext, Dokumentenschema



Einführung in Datenbanksysteme

XML 7.17

Einsatzgebiete für XML

Format zur Repräsentation und zum Austausch semi-strukturierter Daten, z.B. in den Bereichen:

- Modellierung (UML): XMI (XML Metadata Interchange)
- Meta-Daten: RDF (Resource Description Framework)
- Dokumente
- Chemische Formeln
- Business-To-Business Produktionsketten

Ablösen von HTML als der Sprache des Web (mehr Flexibilität)

**Wichtiger
Erfolgsfaktor:
Hohe Marktpräsenz**

Einführung in Datenbanksysteme

XML 7.18

Datendefinition

Instanzebene: XML Dokumente

wohl-geformtes XML Dokument (*well-formed*): Jedes Element (außer der Wurzel des Dokumentes) muss vollständig von einem anderen Element eingeschlossen sein. Das Tag-"Klammergebirge" muss vollständig sein.

gültiges XML Dokument (*valid*): Das Dokument entspricht festgelegten Regeln für den Aufbau des Dokuments (Document Type Definition).

Strukturebene/Typebene: DTDs (Document Type Definition)

- Definition zulässiger Elemente und ihrer Attribute in einer speziellen DTD-Syntax
- Definition der zulässigen Strukturierung und Kombination dieser Elemente

DTD-Syntax: Typkonstruktoren

Einziges Basisdatentyp ist Text (in XML #PCDATA)

Verfügbare Konstruktoren

Aggregation	(Titel, Budget)
Kollektion	(Projekt*) und (Abteilung+)
Varianten	(Budget Kostenträger)
Option	(Budget?)

PCDATA =
Parsable
Character Data

Diese Konstruktoren können orthogonal miteinander kombiniert werden

Beispiel-Definitionen zulässiger Elemente:

```
<!ELEMENT Projekte (Projekt*)>
<!ELEMENT Projekt (Titel, Budget? )>
<!ELEMENT Titel (#PCDATA)>
```

Besondere Elementdefinitionen:

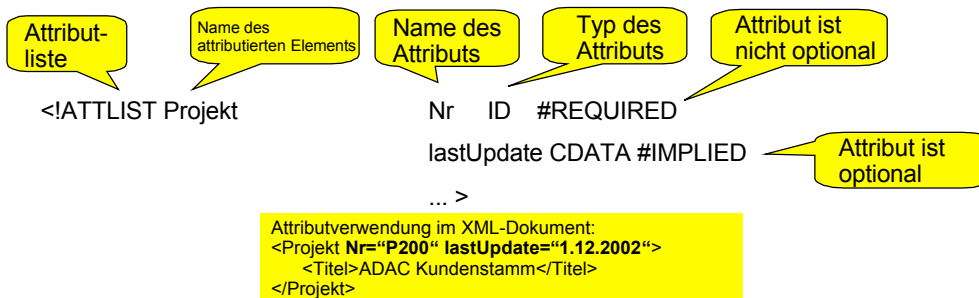
ANY	beliebiger Inhalt (andere Elemente und Text), z.B. <!ELEMENT Projektbeschreibung ANY>
EMPTY	das Element ist leer (kein Inhalt), z.B. für Elemente, die als Flags benutzt werden <!ELEMENT ist_EU_Projekt EMPTY>

Attribute und Attributdeklaration

Deklaration einer Liste von Attributen zu einem Element in der DTD.

Die einzelne Attributdefinition besteht aus

- Name des Attributs
- Typ des Attributs (siehe nächste Folie)
- Angabe, ob das Attribut optional ist (#REQUIRED, #IMPLIED)
- Default-Wert für das Attribut (optional)



Attributtypen (Auswahl)

CDATA	beliebige Zeichenkette
ID	Identifikator, eindeutig innerhalb eines Dokuments (vgl. Schlüsselattribut)
IDREF	Referenz auf ein Element, das ein ID-Attribut besitzt (vgl. Fremdschlüsselattribut)
NMTOKEN	Zeichenkette mit Einschränkungen (darf z.B. keine Leerzeichen enthalten)
ENTITY	benanntes Dokumentfragment

Typen für **mengenwertige** Attribute:

NMTOKENS, ENTITIES, IDREFS

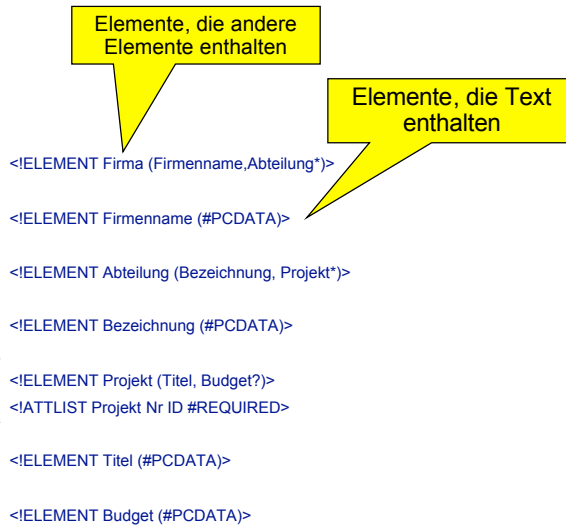
Document Type Definition (DTD)

Eigene Syntax zur Definition gültiger Dokumentstrukturen.

DTD: vgl. Grammatik einer formalen Sprache

Dokument: Ein Satz in der durch die DTD definierten Sprache

Mit Attributen versehene Elemente



Attribute vs. Elemente

```

<Projekt>
  <Nr>P100</Nr>
  <Titel>DB Fahrpläne</Titel>
  <Budget>300000</Budget>
</Projekt>
    
```

Element-Lösung

- Daten befinden sich im Content-Teil des Elements („zwischen den Tags“)
- geschachtelte (auch rekursive) Strukturen möglich
- sehr eingeschränkte Typisierung



```

<Projekt Nr="P100"
  Titel="DB Fahrpläne"
  Budget="300000">
</Projekt>
    
```

Attribut-Lösung

- Daten sind Bestandteil des Start-Tags
- Attributwerte sind immer „flach“
- sind typisiert (ID, IDREF, CDATA etc.)
- Parser kann Attributwerte interpretieren (z.B. Existenz einer per IDREF referenzierten ID prüfen)

Mischformen möglich (vgl 7.15)

Darstellung von Beziehungen durch Wertstrukturen

1:N Beziehungen (Hierarchien, Kopiersemantik):

Schachtelung von XML-Elementen in andere XML-Elemente

Beispiel (Beziehung Projekte einer Abteilung)

<!ELEMENT Abteilung (Name, Projekt*)>

In der DTD

```
<Abteilung>
  <Name>Unix Software</Name>
  <Projekt Nr="P100">
    <Titel>DB Fahrplaene</Titel>
    <Budget>300000</Budget>
  </Projekt>

  <Projekt Nr="P200">
    <Titel>ADAC Kundenstamm</Titel>
  </Projekt>
...
</Abteilung>
```

Im XML Dokument

Elementschachtelung ist nicht für komplexere Beziehungen (N:M, k-när) geeignet (Redundanz vgl. HDM)

Darstellung von Beziehungen durch Referenzen

Darstellung über IDs und IDREFS (Referenzsemantik)

Beispiel N:M-Beziehung zwischen Abteilungen und Projekten:

```
<!ATTLIST Abteilung   Kurz ID           #REQUIRED
                   Projekte IDREFS      #IMPLIED>
<!ATTLIST Projekt     Nr ID           #REQUIRED
                   Abteilungen IDREFS   #IMPLIED>
```

In der DTD

```
<Abteilung Kurz = "UXSW" Projekte = "P100 P200"> ... </Abteilung>
<Projekt Nr = "P100" Abteilungen = "UXSW MFSW LTSW" > ...</Projekt>
```

Im XML Dokument

Voraussetzung: Referenzierendes und referenziertes Element sind beide Teil desselben Dokuments.

Eindeutigkeit der Identifikatoren und Existenz von referenzierten Identifikatoren (referentielle Integrität) werden vom XML Parser überprüft.

Nachteile:

- IDREFs können nur IDs im selben Dokument referenzieren
- ID-Werte müssen eindeutig sein, unabhängig vom XML-Element, in dem sie verwendet werden

Alternativen zur Darstellung von Beziehungen

a) Verwendung separater Elemente

```
<Projekt Nr="P100">
  <Titel>DB Fahrplaene</Titel>
  <Budget>300000</Budget>
  <Betreuung Abteilung="MFSW"/>
  <Betreuung Abteilung="UXSW"/>
</Projekt>
```

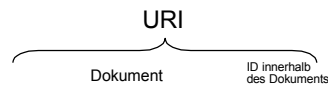
```
<!ELEMENT Projekt (Titel, Budget, Betreuung*)>
<!ATTLIST Projekt Nr ID #REQUIRED>
<!ELEMENT Betreuung EMPTY>
<!ATTLIST Betreuung Abteilung IDREF #REQUIRED>
```

Anzahl und Struktur der referenzierenden Elemente kann in DTD näher spezifiziert werden

b) Verwendung eines alternativen Link-Mechanismus

(XLink, vgl. 7.29, gestattet z.B. Verweise auf Elemente anderer Dokumente)

```
<Projekt Nr="P100">
  <Titel>DB Fahrplaene</Titel>
  <Budget>300000</Budget>
  <Betreuung xlink:type="simple" xlink:href="http://.../abteilungen.xml|MFSW" />
  <Betreuung xlink:type="simple" xlink:href="http://.../abteilungen.xml|UXSW" />
</Projekt>
```



Referenz kann nur durch einen Xlink-fähigen XML-Parser geprüft werden, der die speziellen XLink-Attributwerte verarbeiten kann.

XPointer

XPointer und *XLink* sind Spezifikationen, die die Markierung, Auswahl (Selektion) und Verknüpfung von Dokumenten und Dokumententeilen definieren.

Die *XPointer* Spezifikation bietet eine bequeme und leicht verständliche Art, *Positionen innerhalb* von XML Dokumenten zu beschreiben:

- Zugriff auf Positionen innerhalb von Dokumenten; auch auf solche, für die der Autor des Dokuments zunächst keinen Ankerpunkt vorgesehen hat,
- exakte Adressierung von Elementen in XML-Dokumenten unter Ausnutzung von Hierarchiebeziehungen (siehe auch XQL/XSL), Positions- und Bereichsmarkierungen,
- eine einfache, für Menschen verständliche Syntax zur Beschreibung von Positionen und Hierarchiebeziehungen zwischen Elementen in Dokumenten.
- Beispiel: /1/5/3 bestimmt das fünfte Element (/5) in der Folge der Kinder des Wurzelements (/1) und selektiert dessen drittes Kindelement.

XLink

XLink (XML Linking Language) bietet weitergehende Möglichkeiten, ganze Dokumente und Teile von Dokumenten zu verknüpfen. XLink definiert dafür höhere Konzepte als die XPointer Spezifikation.

- mehrwertige Links (1:n, m:n Links),
- Bidirektionale Links (Links, die ähnlich den *relationships* in objektorientierten Datenbanken in beide Richtungen traversierbar sind),
- Links zum Verknüpfen von *read-only* Dokumenten. Damit können von Dritten Links in Dokumente eingefügt werden, ohne dass diese Rechte zum Ändern des Dokuments haben,
- Annotation von Rollen und Beschreibungen an Links (siehe auch annotierte Assoziationen in UML),
- Links können dazu benutzt werden, Datenbankinhalte direkt zu verknüpfen.

Umsetzungen von *XLink* können auf *XPointer* basieren, die beiden Spezifikationen sind aber eigentlich unabhängig voneinander.

XML Schemata (1)

DTDs haben folgende Nachteile:

- DTDs werden nicht in XML notiert. Für das Schreiben von XML Dokumenten und XML DTDs müssen unterschiedliche Werkzeuge verwendet werden.
- DTDs sind in ihrer Ausdrucksmächtigkeit beschränkt, besonders bei der Beschreibung von Datentypen (z.B. *Integer*, *Date*, *Time*, ...)
- Es gibt keine Vererbungsbeziehungen zwischen Elementen einer oder mehrerer DTDs

XML-Schema-Standard:

- Bietet eine Obermenge der Modellierungsmöglichkeiten des DTD-Standards.
- XML-Schemata sind selber XML-Dokumente (keine separate Syntax wie bei DTD)
- Vorteil: XML-Schemata können mit den gleichen Tools erzeugt, geprüft und weiterverarbeitet werden, die auch für „gewöhnliche“ XML-Dokumente zum Einsatz kommen

XML Schemata (2)

XML Schemata werden (im Gegensatz zu DTDs) in XML selbst notiert.

- ❑ Bekannte Notation; Werkzeuge zum Bearbeiten von XML Dokumenten und Schemata.
- ❑ Schemata können wiederum durch (höhere) Schemata beschrieben werden (❑ Meta-Schemata).

In XML Schemata existieren Basisdatentypen wie *int*, *boolean*, *DateTime*, etc. (vgl. <http://www.w3.org/TR/xmlschema-2/#built-in-primitive-datatypes>)

- ❑ Ausgehend davon können anwendungsspezifische Datentypen (*Kontonummer*, *Autokennzeichen*, ...) definiert werden.

Aus vorhandenen Datentypen können neue Datentypen abgeleitet werden:

- ❑ Typhierarchien werden möglich
- ❑ Schemata können Datentypen exportieren und aus anderen Schemata importieren (sie definieren ein *Interface* für andere XML Schemata). ❑ Kombinieren von bestehenden Datentypen in einem neuen Schema.

DTD vs. XML-Schema (Beispieldaten)

<pre> <Firma> <Firmenname>DB-Soft</Firmenname> <Abteilung> <Bezeichnung>Unix Software</Bezeichnung> <Projekt Nr="P100"> <Titel>DB Fahrplaene</Titel> <Budget>300000</Budget> </Projekt> <Projekt Nr="P200"> <Titel>ADAC Kundenstamm</Titel> </Projekt> </Abteilung> <Abteilung> <Bezeichnung>Mainframe Software</Bezeichnung> <Projekt Nr="P300"> <Titel>Telekom Statistik</Titel> <Budget>200000</Budget> </Projekt> </Abteilung> </Firma> </pre>	<pre> <!ELEMENT Firma (Firmenname,Abteilung*)> <!ELEMENT Firmenname (#PCDATA)> <!ELEMENT Abteilung (Bezeichnung, Projekt*)> <!ELEMENT Bezeichnung (#PCDATA)> <!ELEMENT Projekt (Titel, Budget?)> <!ATTLIST Projekt Nr ID #REQUIRED> <!ELEMENT Titel (#PCDATA)> <!ELEMENT Budget (#PCDATA)> </pre>
XML-Dokument	DTD

DTD vs. XML-Schema (Abbildung)

<ELEMENT Firma (Firmenname,Abteilung)>	<pre><xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"> <xs:element name="Firma"> <xs:complexType> <xs:sequence> <xs:element ref="Firmenname"/> <xs:element ref="Abteilung" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element></pre>
<ELEMENT Firmenname (#PCDATA)> <ELEMENT Abteilung (Bezeichnung, Projekt)>	<pre> <xs:element name="Firmenname" type="xs:string"/> <xs:element name="Abteilung"> <xs:complexType> <xs:sequence> <xs:element ref="Bezeichnung"/> <xs:element ref="Projekt" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType> </xs:element></pre>
<ELEMENT Bezeichnung (#PCDATA)> <ELEMENT Projekt (Titel, Budget?)>	<pre> <xs:element name="Bezeichnung" type="xs:string"/> <xs:element name="Projekt"> <xs:complexType> <xs:sequence> <xs:element ref="Titel"/> <xs:element ref="Budget" minOccurs="0"/> </xs:sequence> </xs:complexType> </xs:element></pre>
<!ATTLIST Projekt Nr ID #REQUIRED>	<pre> <xs:attribute name="Nr" type="xs:ID" use="required"/> </xs:complexType> </xs:element></pre>
<ELEMENT Titel (#PCDATA)> <ELEMENT Budget (#PCDATA)>	<pre> <xs:element name="Titel" type="xs:string"/> <xs:element name="Budget" type="xs:string"/> </xs:schema></pre>

Namespaces (1)

Bei Benutzung gleichnamiger Elemente verschiedener DTDs in einem XML Dokument kann es zu **Namenskollisionen** kommen. Diese Problematik ergibt sich fast immer, wenn Daten **verschiedener Anwendungsdomänen** in XML Dokumenten auftreten.

Beispiel: Bibliotheks - Buchbestellung über Internet:

In einem Formular zur Buchbestellung kommt sowohl der Titel der Web-Seite, als auch der Titel des Buches vor. Beide Anwendungsbereiche verwenden denselben Elementnamen, *title*.

```
<html>
  <head><title>TUHH Bibliothek 2000</title></head>
  <body>
    <p>Das Buch
      <book>
        <title>Monty Python's Complete Database Handbook</title>
        <author>Cleese et al.</author>
      </book>
    <p>wurde für Sie vorbestellt.</p>
  </body>
</html>
```

Namespaces (2)

Namespaces wurden in XML eingeführt, um solche Namenskollisionen zu vermeiden. Sie werden mit dem reservierten Attribut *xmlns* deklariert.

```
<html xmlns:html='http://www.w3.org/TR/REC-html40'
      xmlns:book='urn:loc.gov:books' >

<head><html:title>TUHH Bibliothek</html:title></head>
<body>
  <p>Das Buch
    <book>
      <book:title>Monty Python's Complete Database Handbook</book:title>
      <author>Cleese et al.</author>
    </book>
  <p>wurde reserviert.</p>
</body>
</html>
```

Mehrdeutige Elementnamen müssen durch Präfixe eindeutig identifiziert werden.

Zu beachten: Die URIs der Namespaces definieren **keinerlei Semantik** der Elementnamen. Manche Parser akzeptieren nur bestimmte URIs für bekannte Präfixe, wie z.B. *html*, *xsl*.

XML-Anfragesprachen

- Standardisierte Anfragesprachen für XML-Daten existieren noch nicht
- Ein W3C-Standard für XML Query befindet sich in Beratung
- Anforderungen des W3C an eine solche Anfragesprache:
 - Deklarativität (vgl. SQL)
 - Selektion von XML-Elementen anhand
 - ihres Namens,
 - ihrer Attributwerte,
 - ihrer Inhalte
 - Universelle und existentielle Quantifizierung
 - Operationen auf Sequenzen und Hierarchien
 - Kombination der Daten zu neuen Strukturen
 - Aggregationsoperationen (Summenbildung, Elementhäufigkeiten)
 - Sortierung
 - Komposition von Operationen
 - Verfolgung von Referenzen innerhalb und zwischen Dokumenten