

## XML Grundlagen Teil II

Erik Wilde

27.3.2006

<http://dret.net/lectures/xml-fhnw-ss06/>

27.3.2006

XML Vorlesung FHA SS 2006

1

## Übersicht

- Datenmodellierung und XML
- Entities in XML
  - Parameter Entities für die Datenmodellierung
- XML und Zeichensätze

27.3.2006

XML Vorlesung FHA SS 2006

2

## Datenmodellierung und XML

- nicht grundlegend anders als andere Modelle
  - Modellierung in einem abstrakten Modell
  - Anwendung der DTD-spezifischen Mechanismen
- Datenmodellierung als Kern einer Applikation
  - schlechte Modelle ergeben schlechte Applikationen
  - schlechte Modelle erschweren die Programmierung
  - gute Modelle sind einfach zu verstehen
  - gute Modelle sind einfach zu erweitern
- Ideen des *Extreme Programming* verwenden
  - kurze Zyklen von Entwicklung und Feedback
  - inkrementell entwickelte Datenmodelle

27.3.2006

XML Vorlesung FHA SS 2006

3

## Datenmodellierung für Datenbanken

- Modellierung unterscheidet 3 Ebenen
  - *conceptual model*: ER-Modellierung
  - *logical model*: Abbildung auf Datenbanktabellen
  - *physical model*: Implementierung und Optimierung
- XML braucht u.U. andere Mechanismen
  - ER-Modelle sind für Tabellen als Implementierung
  - ER-Modelle können auf XML abgebildet werden
    - Tabellen als dreistufige Hierarchie
    - Schlüssel und Referenzen als ID/IDREF(S) Attribute
  - XML ist mächtiger als das ER-Datenmodell

27.3.2006

XML Vorlesung FHA SS 2006

4

## Gute Datenmodellierung

- Datenmodelle sind Sichten der Welt
  - Reduktion auf einen Ausschnitt (Anwendungsgebiet)
  - Reduktion auf bestimmte Aspekte (Formalismus)
  - Reduktion auf eine Anwendung (Verarbeitung)
- Erfolg des relationalen Modells
  - weniger spezifisch als hierarchische Modelle
  - Zusammenhänge zwischen logischen Einheiten
    - aber auch die Auswahl der Entities ist subjektiv!
- XML beruht auf einem hierarchischen Modell
  - implizit eine "Bewertung" der Daten
  - problematisch für datenorientierte Anwendungen

27.3.2006

XML Vorlesung FHA SS 2006

5

## XML ist hierarchisch

- Erbe aus der Dokumentenwelt
  - Dokumente sind recht klar hierarchisch aufgebaut
  - bei modularem Aufbau schon weniger klar
    - `<!ELEMENT section (content, chapter+)`
    - `<!ELEMENT chapter (content, subchapter+)`
    - `<!ELEMENT part (content, part+)`
- relational und hierarchisch gemischt
  - Entities als eigenständige Teile modellieren
  - klar hierarchische Teile als XML-Strukturen
- auch hier sind Entscheidungen nötig
  - welche Teile sollen wiederverwendbar sein?

27.3.2006

XML Vorlesung FHA SS 2006

6

## Datenmodellierung für XML

- herkömmliche Modelle sind limitiert
  - ER-Modelle decken nur einen Teil ab
- bisher keine Sprache für XML Modellierung
  - spezialisierte Sprachen für einzelne Anwendungen
  - keine generelles *conceptual modeling* für XML
  - *logical models* für XML sind Schemasprachen
    - DTDs als XML 1.0 Mechanismus
    - XML Schema als neuer und komplexerer Standard
  - *physical models* hängen von der Speicherung ab
    - verschiedene Speichermodelle für DOM Bäume
    - Indexierung für XML Datenbanken
    - Abbildung auf andere Strukturen (RDBMS, File System, ...)

27.3.2006

XML Vorlesung FHA SS 2006

7

## Datenmodellierung für DTDs

- strukturiertes Vorgehen in mehreren Schritten
  - anwendungsorientierte Datenmodellierung
  - XML-spezifische Aspekte berücksichtigen
    - hierarchische Strukturen (u.U. Rekursion erlauben)
    - Reihenfolge von Elementen ist signifikant
  - DTD-spezifische Aspekte berücksichtigen
    - sehr schwache Typunterstützung
    - referentielle Integrität nur global möglich mit ID/IDREF(S)
    - keine Co-Constraints
  - DTD-Probleme erkennen und dokumentieren
    - Lösung in einer besseren/ergänzenden Schemasprache
    - Lösung in der Applikation

27.3.2006

XML Vorlesung FHA SS 2006

8

## Design Patterns für DTDs

- DTDs unterstützen Wiederverwendung nicht
  - viel Copy&Paste bei DTD-Design
  - erzeugt Inkonsistenzen bei folgenden Änderungen
- Abhilfe durch Parameter Entities
  - Wiederverwendetes als Parameter Entity definieren
  - Parameter Entity referenzieren
- sehr empfehlenswert für ernsthafte DTDs
  - Benutzung unterliegt der Selbstdisziplin
  - von vielen Tools nicht gut unterstützt

27.3.2006

XML Vorlesung FHA SS 2006

9

## Umgang mit Datenmodellen

- formale Datenmodelle sind immer simplifiziert
  - der Formalismus ist beschränkt
  - nicht alles wird formalisiert
  - nicht alles kann formalisiert werden
- Datenmodelle sollten deklarativ sein
  - besser nachvollziehbar und erweiterbar
  - weniger abhängig von Implementierungsfragen
- Design übersichtlich und erweiterbar halten
  - nachträgliche Änderungen im Rahmen des Formalismus
  - Ergänzungen ausserhalb des Formalismus

27.3.2006

XML Vorlesung FHA SS 2006

10

## Parameter Entities in DTD (XHTML)

```

<!ENTITY % Length "CDATA">
<!ENTITY % Pixels "CDATA">

<!ENTITY % coreattrs
  "id ID #IMPLIED
  class CDATA #IMPLIED
  style %StyleSheet; #IMPLIED
  title %Text; #IMPLIED">

<!ENTITY % attrs "%coreattrs; %l18n; %events;">

<!ELEMENT hr EMPTY>

<!ATTLIST hr
  %attrs;
  align (left | center | right) #IMPLIED
  noshade (noshade) #IMPLIED
  size %Pixels; #IMPLIED
  width %Length; #IMPLIED >

```

27.3.2006

XML Vorlesung FHA SS 2006

11

## Entities

- das grundlegende Konstrukt zur Strukturierung
  - jedes "Stück XML Text" ist konzeptionell ein Entity
  - Entities werden deklariert (wie Elemente/Attribute)
- Benutzung in XML über speziellen Markup
  - benutzt durch eine *entity reference* (&entity;)
  - XML Prozessor fügt den *replacement text* ein
- verschiedene Entity-Typen
  - *Character* (zur Darstellung beliebiger Unicode-Zeichen)
  - *Parameter* (nur in DTDs verwendet, z.B. %attributes;)
  - *Internal General* ("Text-Makros" in Dokumenten, z.B. &uuml;)
  - *External Parsed General* (externe XML Ressourcen)
  - *Unparsed* (externe nicht-XML Ressourcen)
- vordefinierte Entities für Markup-Zeichen (nur 5!)

27.3.2006

XML Vorlesung FHA SS 2006

12

## Parameter Entities in DTDs

- Text-Makros für DTDs
  - gleicher Zweck wie *Internal General Parsed* (&xx;)
  - keine DTD-Semantik, reine Ersetzungsfunktion
- gut definierte Makros repräsentieren Semantik
  - ausschliesslich der Eigendisziplin überlassen
  - keine Tools für Konsistenz-Überprüfungen
- Erhalten von Informationen für später
  - selbstdokumentierender Code
  - Weiterentwicklung der DTD
  - Weiterverwendung durch andere
  - Verwendung eines mächtigeren Mechanismus

27.3.2006

XML Vorlesung FHA SS 2006

13

## Attributlisten für Elementmengen

- in XML nicht unterstützt
  - kein inhaltlicher Zusammenhang von Attributlisten
  - Zusammensetzen mit Parameter Entities (XHTML)
- beim Umstieg auf SGML
  - SGML unterstützt Attributlisten für Elementlisten
  - `<!ATTLIST (a|b|c) ... >`
  - aber keine modular aufgebauten Listen
  - `<!ATTLIST (a|b|c) %core; %i18n; >`
- beim Umstieg auf XML Schema
  - unterstützt Attributlisten für Elementlisten
  - unterstützt modular aufgebaute Listen

27.3.2006

XML Vorlesung FHA SS 2006

14

## Strukturierung von DTDs

- DTDs sind nicht immer in einem Dokument
  - DTD wird vom XML Dokument referenziert
  - DTD referenziert weitere Teile
- modularer Aufbau von DTDs
  - bessere Übersicht über die DTD
  - Wiederverwendung von Teilen für verschiedene DTDs
  - konfigurierbare DTDs
- bei kleineren Projekten nicht unbedingt nötig
  - für grosse Projekte ein wichtiger Designschritt
  - sorgfältige Planung vor der Definition der DTDs

27.3.2006

XML Vorlesung FHA SS 2006

15

## External und Internal Subset

- DTDs können internal oder external sein
  - intern sind sie integraler Teil des Dokuments
  - extern werden sie per Identifier referenziert
- beide Mechanismen können kombiniert werden
  - *Internal Subset* wird als erstes interpretiert
  - internal kann external überschreiben
- im Allgemeinen wird external verwendet
  - Wiederverwendbarkeit von Dokumenten
  - Grösse von Dokumenten
  - internal u.U. für spezifische Erweiterungen

27.3.2006

XML Vorlesung FHA SS 2006

16

## Dokumente und DTDs (Beispiele)

```
<?xml version="1.0"?>
<!DOCTYPE greeting SYSTEM "hello.dtd">
<greeting>Hello, world!</greeting>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE greeting [
  <!ELEMENT greeting (#PCDATA)>
]>
<greeting>Hello, world!</greeting>
```

27.3.2006

XML Vorlesung FHA SS 2006

17

## Conditional Sections

- Teile von DTDs können variabel bleiben
  - Teile können ignoriert werden
    - `<![IGNORE[ declarations ]]>`
  - Teile können interpretiert werden
    - `<![INCLUDE[ declarations ]]>`
- Steuerung über Entities ist sinnvoll
  - `<![%switch;[ declarations ]]>`
- verbreitetes Design Pattern
  - Switches im *Internal Subset* deklarieren
  - das *External Subset* wird auf diese Weise konfiguriert

27.3.2006

XML Vorlesung FHA SS 2006

18

## NOTATION Declarations

- Format: `<!NOTATION PNG SYSTEM "png">`
- deklarieren Werte von NOTATION Attributen
  - für die Interpretation von Elementen
- deklarieren das Format von *unparsed Entities*
  - `<!ENTITY ich SYSTEM "portrait.png" PNG>`
    - diese Entities dürfen nur in Attributwerten referenziert werden
  - `<!ATTLIST portrait pic ENTITY #IMPLIED >`
  - `<portrait pic="ich"/>`

27.3.2006

XML Vorlesung FHA SS 2006

19

## CDATA Sections

- Markup benötigt bestimmte Zeichen
  - können durch Entities ersetzt werden: `&amp;` & `&lt;`;
- CDATA Sections werden benutzt
  - falls sehr viele Markup-Zeichen vorkommen
  - falls nicht sicher ist, was vorkommen wird (Includes)
- CDATA benutzt ebenfalls Markup
  - beginnt immer mit der Zeichenkette `<![CDATA[`
  - endet immer mit der Zeichenkette `]]>`
    - `]]>` als Inhalt einer CDATA Section ist nicht möglich
  - Schachtelung ist nicht erlaubt
- Markup (auch Entities) wird nicht interpretiert

27.3.2006

XML Vorlesung FHA SS 2006

20

## Beispiel einer CDATA Section

```
<?xml version="1.0" encoding="UTF-8"?>
<document>
  <element>CDATA sections <![CDATA[
<document><element></element></document>
]]> werden nicht als Markup
interpretiert.</element>
</document>
```

27.3.2006

XML Vorlesung FHA SS 2006

21

## "Character Sets"

- der Begriff sollte vermieden werden!
- besser definierte Begriffe sind
  - *Character Repertoire* (Menge an Zeichen)
  - *Character Code* (Definition von *Code Points*)
  - *Character Encoding* (Codierung für *Code Points*)
- 7-bit ASCII (ISO 646) definiert alles zusammen
- ISO 8859 ist ASCII auf 8-bit erweitert
  - Varianten für verschiedene Erweiterungen (ISO 8859-1 usw.)
- Unicode trennt die verschiedenen Schritte
  - ISO 10646 und Unicode sind harmonisiert
  - Unicode definiert verschiedene Encodings

27.3.2006

XML Vorlesung FHA SS 2006

22

## XML und Zeichensätze

- allgemeines Problem in der Informatik
  - im W3C bearbeitet im Bereich "I18N"
  - verbreitete Standards sind ASCII und ISO 8859-1
  - Benachteiligung vieler anderer Sprachen
- XML ist ein zeichenbasiertes Format
  - Problem bekannt von HTML (Umlaute)
  - XML verwendet Unicode als Default
- Unicode legt eine Menge von Zeichen fest
  - Zeichen identifiziert über *Code Points*
  - ein spezifisches Encoding legt die binäre Form fest
  - meistens UTF-8 (jedes ASCII-Dokument ist UTF-8!)

27.3.2006

XML Vorlesung FHA SS 2006

23

## XML und die Zukunft

- momentan gültig ist XML 1.0 Third Edition
  - korrigiert Fehler des XML 1.0 Standards
  - fügt keine neuen Features dazu
- neueste Version von XML ist XML 1.1
  - keine revolutionären Erweiterungen
  - verbesserter Umgang mit exotischen Zeichensätzen
  - Anpassungen an Grossrechnerumgebungen
- momentan keine weitere Aktivitäten
  - unverbindliche Prognose: XML 2.0 enthält XML Namespaces, XML Infoset, XML Base (keine DTD)

27.3.2006

XML Vorlesung FHA SS 2006

24

## Zusammenfassung

- Datenmodellierung und XML
  - Besonderheiten von XML-basierten Datenmodellen
- Entities
  - Referenzen auf Stücke von XML Text
  - Internal vs. External und Parameter vs. General
- XML ist ein zeichenbasiertes Format
  - CDATA Sections
  - Fragen nach Character Repertoire und Encoding
  - XML verwendet als Default Unicode/UTF-8