Informations- und Datenmodelle und XML

Erik Wilde
Institut für Technische Informatik und Kommunikationsnetze (TIK)
ETH Zürich

Überblick

- XML 1.0 Syntax und Modell
- Informations- und Datenmodelle
- Probleme bei der Verarbeitung
  - CDATA Sections
  - Elemente vs. Attribute
- Lösungsansätze
- Zusammenfassung

XML 1.0

- XML 1.0 definiert zwei Syntaxen
  - beide definiert auf der Zeichenebene
  - ergänzt um Bedeutungsbeschreibungen
- XML 1.0 Document Syntax
  - Grammatik mit Constraints
  - element ::= EmptyElemTag | STag content ETag [WFC: Element Type Match] [VC: Element Valid]
  - an einigen Stellen wird interpretiert
    - Normalisierung von Attributwerten
    - ID/IDREF-Attribute (falls DTD vorhanden)
- XML 1.0 DTDs
  - Constraints für XML Dokumente
- XML Informationsmodell ist vage definiert

Informations- und Datenmodelle

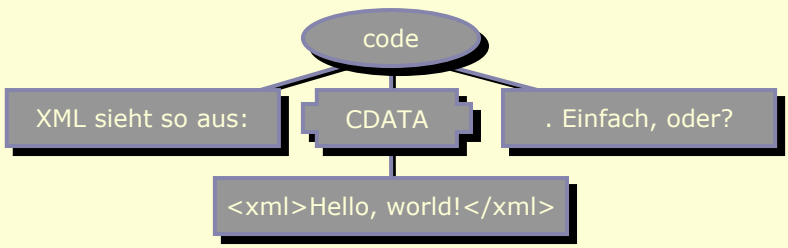
- RFC 3444 definiert zwei Ebenen
- Informationsmodelle
  - dienen der Modellierung
  - sind abstrakte Gebilde
  - bieten keine Schnittstellen zum Umgang
- Datenmodelle
  - implementieren ein Informationsmodell
  - bieten eine konkrete Schnittstelle
- XML 1.0 ist ein Datenmodell
  - es existiert kein verbindliches Informationsmodell
  - DOM/SAX/XPath/... sind alternative Datenmodelle
    - allerdings mit gewissen Einschränkungen...

XML Information Set

- entstanden aus "Leidensdruck"
  - immer wieder Beantwortung der gleichen Fragen
  - "willkürliche" Festlegung der Antworten
- definiert kein Datenmodell!
- Pflichtlektüre für XML-Modellierer
- http://www.w3.org/TR/xml-infoset/#omitted
  - leere Elemente: <wert/> vs. <wert></wert>
  - verschiedene Attribut-Informationen
    - Quotes: attr="wert" vs. attr='wert'
    - Reihenfolge der Attribute: a='1' b='2' vs. b='2' a='1'
  - Entity References
    - die gesamte Entity-Struktur des Dokuments
    - Character References vs. Characters: ü vs. ü

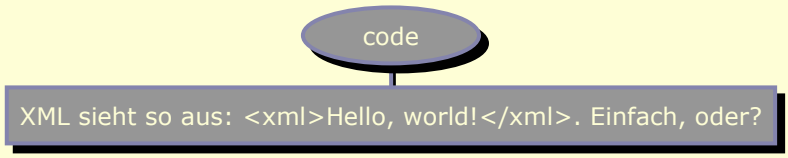
Beispiel 1: CDATA Sections

- CDATA Sections für Markup-Zeichen in XML
  - <code>XML sieht so aus: <![CDATA[<xml>Hello, world!</xml>]]>. Einfach, oder?</code>
  - sehr nützlich für XML (oder HTML) in XML



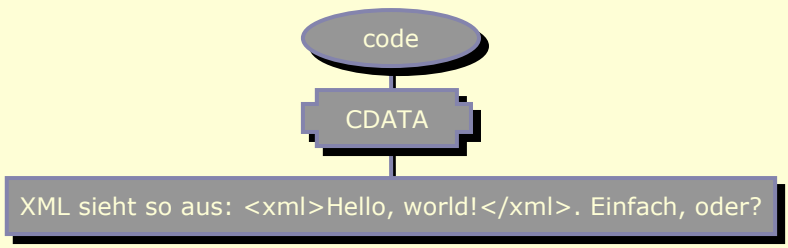
CDATA und XSLT

- XSLT benutzt XPath als Expression Language
  - XPath basiert auf dem XML Information Set
  - das XML Infoset kennt keine CDATA Sections
- CDATA Sections verschwinden in XSLT



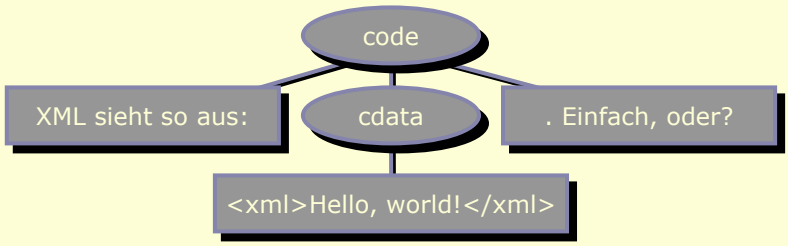
CDATA und XSLT (Fix 1)

- XSLT kann CDATA Sections erzeugen
  - aber nur für komplette Elemente
  - unabhängig von der Existenz in der Eingabe
- Ausgabesteuerung in <xsl:output>
  - <code><![CDATA[XML sieht so aus: <xml>Hello, world!</xml>. Einfach, oder?]]></code>



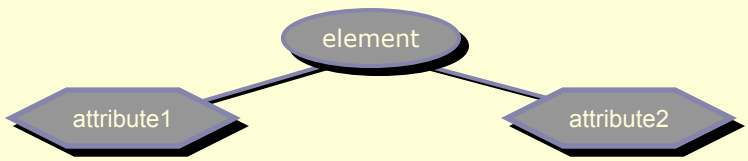
CDATA und XSLT (Fix 2)

- Abbilden auf unterstützte Strukturen
  - <code>XML sieht so aus: <cdata><xml>Hello, world!</xml></cdata>. Einfach, oder?</code>
- nach erfolgter Transformation Rückwandelung
  - z.B. mit einem DOM-basierten Programm



Beispiel 2: Element vs. Attribut

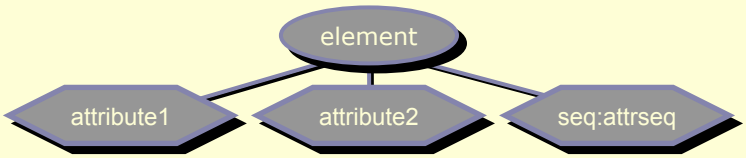
- Attribute in XML haben eine Reihenfolge
  - in vielen Anwendung werden sie als Menge behandelt
  - XSLT kennt die Reihenfolge nicht, DOM auch nicht
- Anwendungen brauchen die Reihenfolge u.U.
  - Frage der Modellierung Elemente vs. Attribute
  - Frage der Bedeutung der Attribute (Liste vs. Menge)



- <element attribute1="A" attribute2="B"/>

Element vs. Attribut (Fix 1)

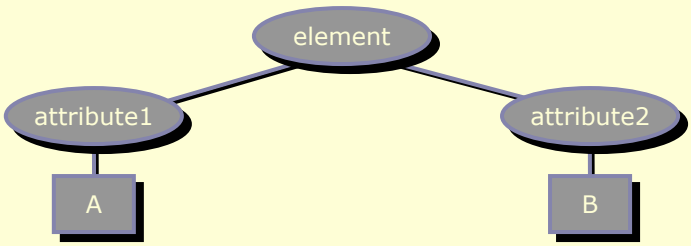
- Attributreihenfolge in Attribut merken
  - z.B. als NMTOKENS Attribut
  - Erzeugung mit einem speziellen API
- <element attribute1="A" attribute2="B" seq:attrseq="attribute1 attribute2"/>



- u.U. per Namespace vom Inhalt trennen

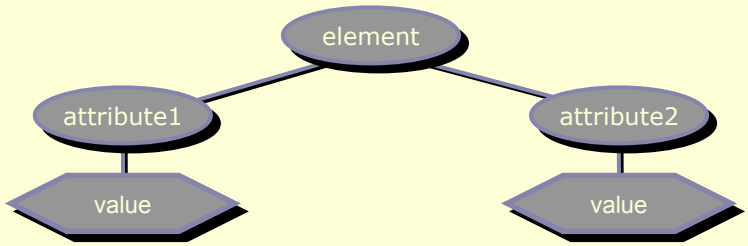
Element vs. Attribut (Fix 2a)

- Attribute als Elemente umformen
  - Reihenfolge von Elementen ist relevant
  - <element><attribute1>A</attribute1> <attribute2>B</attribute2></element>
- Lösung mit Nebenwirkungen
  - Attributwertnormalisierung findet nicht mehr statt



Element vs. Attribut (Fix 2b)

- Attribute als eigene Elemente mit Attributen
  - belässt den Attributen ihren Typ (wichtig bei DTDs)
  - <element><attribute1 value="A"/> <attribute2 value="B"/></element>
- Vermeidung der Nebenwirkungen von Fix 2a



XML Informationsmodelle

- es gibt nicht "das" XML Informationsmodell
  - DOM Node Tree
  - XML Information Set
  - XPath 1.0 Node Tree
  - XML Schema PSVI
  - XPath 2.0 Node Tree
- Wahl des Modelles durch Wahl der Tools
  - Tools sind extrem wichtig für XML Anwendungen
  - Einschränkungen bei sehr guten Gründen zulassen
    - Konsequenzen für alle Beteiligten (Austauschpartner)
- z.T. Wahl der Modelle durch die Plattform
  - z.B. basiert Cocoon auf SAX-Pipelines

Zusammenfassung

- XML hat kein etabliertes Informationsmodell
  - XML Infoset als vom W3C definierte Marke
  - XML Infoset hat deutliche Einschränkungen
- bei XML-Anwendungen strukturiert vorgehen
  1. das benötigte Informationsmodell definieren
  2. die notwendigen XML-Technologien identifizieren
  3. Einschränkungen der Technologien identifizieren
  4. u.U. Anpassung des Informationsmodelles
- XML ist nur eine Syntax!
  - "Your Noise maybe my Signal!"
  - <bits><one/><zero/><zero/><one/><one/>...

Besten Dank

Fragen? Kommentare?

mailto:net.dret@dret.net

http://dret.net/netdret/

http://dret.net/netdret/publications#zgdv2003