X-Technologien

XML and Friends

9. Juli 2001

Jörn Clausen
joern@TechFak.Uni-Bielefeld.DE

Übersicht

XML	Extensible Markup Language
SAX	Simple API for XML
DOM	Document Object Model
XSL	Extensible Stylesheet Language
XPath	XML Path Language
XSLT	XSL Transformations
XSL FO	XSL Formatting Objects

SGML	Standard Generalized Markup Language
DSSSL	Document Style Semantics and Specification Language

Markup – Wofür?

- ursprüngliche Domäne: Textdokumente
- Auszeichnung von *Bedeutung* statt *Formatierung*
- Stylesheets beschreiben Formatierung
- Anwendungen: DocBook, CALS, HTML
- heute: strukturierte Beschreibung von Daten aller Art
- Bioinformatik: BioXML (GAME), MAML, GEML, Pise, . . .

XML – Extensible Markup Language

- ist . . .
  - *keine* Markupsprache
  - *kein* Allheilmittel, trotz des Hypes
  - SGML−−
- ist aber auch . . .
  - einfach zu erlernen
  - weithin akzeptiert
  - durch viele Implementierungen etabliert

ein Beispiel

```
<?xml version="1.0"?>

<presentation>
  <slide>
    <title use="internal">XML &amp; Friends for Dummies</title>
    <title use="external">Doing B2B with XML &amp; Friends</title>
    <ilist>
      <item>XML is not a markup language
        (unlike HTML)</item>
      <item>XML is a meta language for
        defining markup languages</item>
      <item>XML instances can be <emph>well
          formed</emph> or even
        <emph>validating</emph></item>
    </ilist>
  </slide>
</presentation>
```

Aufbau von XML: Elemente

- öffnendes und schließendes *tag*:

  `<item>XML is not a ...</item>`

- keine Minimierungsregeln
- leeres Inhaltsmodell:

  `<hr/>` statt `<hr></hr>` statt `<hr>`

- Schachtelung muß „passen“: *well-formed*

Aufbau von XML: Attribute

- Zusatzinformationen zu Elementen

```
<title use="internal">... for Dummies</title>
```

- Attribute im öffnenden tag
- Wertebereich: Aufzählungstyp, Zahlen, Freitext
- schlechte Typisierung
- Design-Frage: Wann Element, wann Attribut?

```
<date y="2001" m="7" d="9"/>
```

vs.

```
<date><y>2001</y><m>7</m><d>9</d></date>
```

Aufbau von XML: Entitäten

- Makros und Sonderzeichen
- aus HTML bekannt: `&auml;` `&ecirc;` `&copy;`
- in XML vordefiniert: `&` `<` `>` `&apos;` `&quot;`
- XML verwendet Unicode

Document Type Definition

- Grammatik kann durch DTD beschrieben werden
- `<!DOCTYPE presentation SYSTEM "presentation.dtd">`
- Parser kann XML-Instanz gegen DTD *validieren*
- Definition von
  - Elementen: Inhaltsmodell
  - Attributen: Wertebereich, Vorbelegung
  - Entitäten: character entities, parameter entities
- DTD sehr eingeschränkt
- XML Schema, RELAX NG, ca. 10 weitere Vorschläge

DTD zum Beispiel

```
<!ENTITY % text          "(#PCDATA|emph|bold|ital)*">

<!ELEMENT  presentation  (slide)+>
<!ELEMENT  slide         (title*, ilist)>
<!ELEMENT  title         %text;>
<!ATTLIST  title
             use         (internal|external)   #IMPLIED>

<!ELEMENT  ilist         (item)+>
<!ELEMENT  item          %text;>

<!ELEMENT  emph          %text;>
<!ELEMENT  bold          %text;>
<!ELEMENT  ital          %text;>
```

Name Spaces

- Kombination von verschiedenen XML-Sprachen
- z.B. allgemeine, wiederverwendbare Definition von Tabellen:

```
<pr:presentation xmlns:pr="http://www.slides.org/presentation-1.0">
  <pr:slide>
    <pr:title>increasing use of XML</pr:title>
    <cals:table xmlns:cals="http://www.army.mil/SPEC-CALS">
      <cals:tr>
        <cals:th>year</cals:th>
        <cals:th>applications</cals:th>
      </cals:tr>
      <cals:tr>
        <cals:td>1998</cals:td>
        <cals:td>15</cals:td>
      </cals:tr>
```

Verarbeitung von XML

- zwei Verfahren:
  - event-basiert
  - Baumstruktur
- Speicherplatz vs. freier Zugriff
- Geschwindigkeit vs. startup-Zeit
- stream-Fähigkeit

SAX – The Simple API for XML

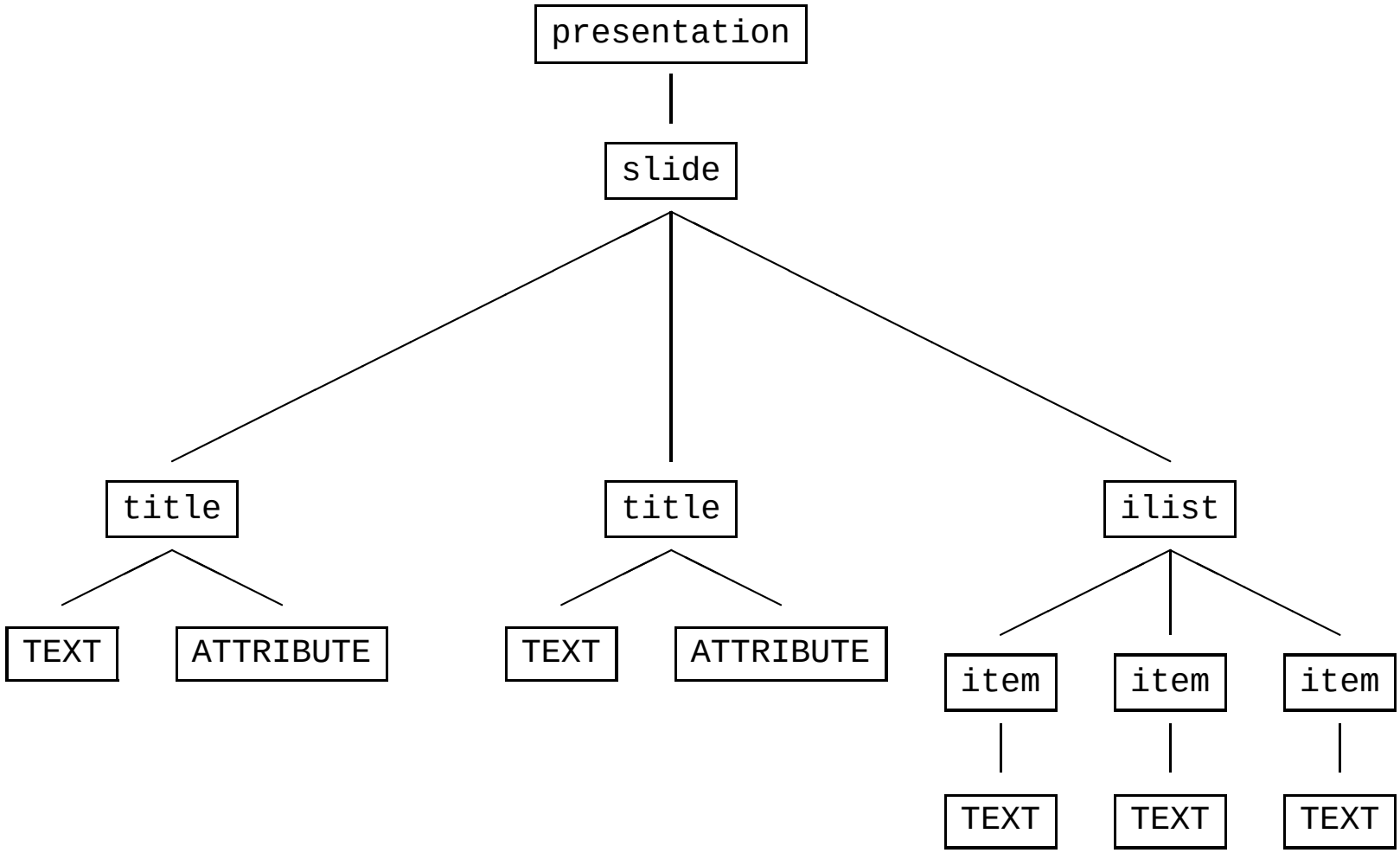
- events: u.a.
  - `startDocument`, `endDocument`
  - `startElement`, `endElement`
  - `characters`
- z.B. einfache Übersetzung nach LaTeX:

`<slide>`	$\longrightarrow$	`\begin{slide}`
`</slide>`	$\longrightarrow$	`\end{slide}`
`<item>`	$\longrightarrow$	`\item`
`</item>`	$\longrightarrow$	
`<bold>`	$\longrightarrow$	`\textbf{`
`</bold>`	$\longrightarrow$	`}`

DOM – Document Object Model

- Dokument als Baum
- Elemente, Attribute, Text usw. sind die Knoten
- Klassen: `Node`, `NodeList`, `Element`, `Text`
- Methoden: `childNodes`, `nextSibling`, `parentNode`, `getElementsByTagName`

das Beispiel als Baum



XSL – Extensible Stylesheet Language

- XML beschreibt Syntax
- Formatierung? Semantik?
- XSL besteht aus mehreren Standards:
  - Zugriff auf bestimmte Knoten (ähnlich DOM)
  - Transformation von XML nach XML
  - XML-Sprache für Layout-Anweisungen

XPath

- enthält
  - Pfadbeschreibung, Bedingungen
  - Funktionen
- *Achsen*: `child`, `parent`, `sibling`, `attribute`
- Verwendung:

`slide`	`slide`-Kind des aktuellen Knotens
`slide/title`	`title`-Element unter `slide`
`slide/title/@use`	`use`-Attribut
`//title[@use="internal"]`	bestimmte `title`-Elemente

Beispiele für XPath

```
juser@hobel> xpath slides.xml ’/presentation/slide/title’
<title use="internal">XML &amp; Friends for Dummies</title>
<title use="external">Doing B2B with XML &amp; Friends</title>

juser@hobel> xpath slides.xml ’/presentation/slide’
<slide>
      <title use="internal">XML &amp; Friends for Dummies</title>
      <title use="external">Doing B2B with XML &amp; Friends</title>

juser@hobel> xpath slides.xml ’//title/@use’
use="internal"
use="external"

juser@hobel> xpath slides.xml ’//title[@use="internal"]’
<title use="internal">XML &amp; Friends for Dummies</title>

juser@hobel> xpath slides.xml ’//item[position()=2]’
<item>XML is a meta language for
          defining markup languages</item>
```

XSLT – XSL Transformations

- Umformung von XML nach
  - XML: Extraktion von Daten, Vorstufe der Formatierung
  - HTML: Visualisierung
- *style sheet* definiert *templates*
- Verarbeitung durch XSLT-Prozessor
- diverse Implementierungen

das Beispiel in HTML

```
<xsl:template match="/presentation">
  <html>
    <body>
      <xsl:apply-templates/>
    </body>
  </html>
</xsl:template>

<xsl:template match="slide">
  <xsl:apply-templates select="title[@use=’external’]"/>
  <xsl:apply-templates select="ilist"/>
</xsl:template>

<xsl:template match="title">
  <h1>
    <xsl:value-of select="."/>
  </h1>
</xsl:template>
```

XSL Formatting Objects

- XML-Sprache zur Beschreibung von Formatierung
- sehr komplex, aber universell
- FO wird durch XSLT-style-sheet erzeugt
- FO wird durch FO-Engine in Ausgabeformat umgewandelt
- FOP (Apache) erzeugt PDF, FrameMaker, PCL, ASCII, Preview

XML-Ausgabe von blast in HTML