XML - eXtensible Markup Language

Einleitung

Seit einiger Zeit wird in den gängigen Internet-Fachzeitschriften häufig von XML berichtet. Ziel dieses Artikels ist es, einen Überblick über die Möglichkeiten und Funktionalitäten von XML zu geben. Aktuelle in der Bearbeitung bzw. in der Definition durch das World Wide Web Consortium befindliche Konzepte wie Namensräume, DOM (Document Object Model) oder Schemes sollten hier nicht im Detail dargestellt werden. Sie werden ggf. ebenso wie die Definition und der Aufbau typischer Dokumenttypdefinitionen zu einem späteren Zeitpunkt erläutert.

Zum Begriff Dokument

Ein Dokument kann durch ein Dokumentenmodell beschrieben werden. Im folgenden Modell wird ein Dokument grundsätzlich als eine Einheit bzw. Mischung aus Inhalt, Struktur und Layout angesehen. Der eigentliche Inhalt des Dokuments enthält die Daten bzw. die zu vermittelnden Informationen in Text, Bild, Ton etc. Die Struktur eines Dokuments spiegelt den Aufbau und die Abfolge der Informationen wider, und das Layout dient der Visualisierung dieser strukturierten Informationen. Dabei beschränkt sich der Begriff Dokument nicht auf eine Datei. Ein Dokument kann aus verschiedenen Dateien bestehen, z.B. aus Textdateien, Graphikdateien, sowie Ton- und Videodateien.

Textverarbeitungsysteme speichern Dokumente in bestimmten Dateiformaten ab. Diese gehen mit dem dargestellten Dokumentenmodell unterschiedlich um. Man unterscheidet daher zwischen Dokumentformaten, die formatierungsorientiert arbeiten, die ein verallgemeinertes Markup[1] benutzen und denen, die ein regelgebundenes Markup benutzen.

Formatierungsorientierte Dokumentformate enthalten eine Mischung aus dem aktuellen Inhalt des Dokuments und der Beschreibung des gewünschten Formats, so arbeiten z.B. Rich Text Format (RTF) oder das LaTeX-Format. Dabei wird der Text in Formatierungsanweisungen eingeschlossen, die Tags oder Codes genannt werden. Oft werden diese Dateiformate durch WYSIWIG-Systeme unterstützt, wie z.B. Microsoft Word.

Beispiel (LaTeX):

```
\begin{document}
Dies ist ein Beispiel f\"ur das Benutzen von
{\bf Formatierungsmarkup}.
Es gibt die Möglichkeit, Zeichen {\it kursiv},
{\bf fett} oder auch {\small klein} oder {\Large
gro\"s}zu setzen.
\end{document}
```

Dokumentformate, die ein verallgemeinertes Markup benutzen, bringen die Formatierungsanweisungen nicht direkt im Inhalt des Dokuments unter. Hier werden Tags gebildet, die eine logische Abfolge der Elemente bzw. den Aufbau eines Dokuments beschreiben. Sie werden auch als semantische Tags bezeichnet. Die Interpretation bzw. Präsentation der Dokumente wird entsprechend der benutzten Tags in Style Sheets oder Formatvorlagen vorgenommen.

Beispiel (XML):

XML-Dokument

```
<body>
<section>Dies ist eine Überschrift </section>
<p> und dies ein Paragraph</p>
<subsection> Eine Unterüberschrift</subsection>
<p> Diese ist ein weiterer Paragraph</p>
</body>
```

Style Sheet (CSS)

```
BODY
{ background-color: #FFFFFA }
section
{ display:block;
  font-weight:bold;
  font-size: 18pt;
  margin-top:1em;
  text-align:center; }
p
{ display:block;
   font-size:12pt;
   font-weight:normal; }
```

Dokumentformate, die ein regelgebundenes Markup benutzen, erfordern, daß die Dokumente nach bestimmten Regeln aufgebaut bzw. erstellt werden, um Dokumente mit Computer-Systemen zuverlässig durchsuchen oder bearbeiten zu lassen. So könnte es zum Beispiel sein, daß ein Gerichtsprotokoll bestimmten Anforderungen genügen, d.h. bestimmte Daten beinhalten, muß. Das sind die Namen des Richters, des Beklagten, beider Anwälte und (optional) die Namen der Geschworenen (soweit vorhanden). Um sicher zu gehen, daß alle benötigten Daten vorhanden sind, müßte der Computer für uns die Regeln, d.h. die Eingabe wichtiger Elemente, erzwingen. Würde der Gerichtsschreiber nun im Protokoll ein Element vergessen, würde das System das Protokoll validieren und feststellen, daß es ungültig ist. Natürlich existieren weitere Arten von Dokumenten. Diese besitzen eine andere Struktur, wie z.B. ein Testament oder eine Rechnung. Dann muß

1 Markup bedeutet hier die Auszeichnung von Dokumenten, insbesondere von Texten.

für jeden Typ eines Dokuments definiert werden, was gültig (für diesen Typ) heißt. In der SGML-Terminologie sind das Dokumenttypen, und die formale Definition, die jeden Typ beschreibt, wird als Dokumenttypdefinition (DTD) bezeichnet (nach [2], Seite35f).

Beispiel (XML): (aus [2], Seite 84f)

DTD: greeting.dtd

```
<!ELEMENT greeting (salutation, addressee)>
        <!ELEMENT salutation (#PCDATA) >
<!ELEMENT addressee (#PCDATA)>
```

Dokument greeting.xml

```
<?xml version="1.0"?> <!DOCTYPE greetings
        SYSTEM file://greeting.dtd" >
<greeting>
<salutation> Hello </salutation>
<addressee> World </addressee></greeting>
```

Was ist XML?

XML wird oft auch als SGML Light bezeichnet. Das liegt daran, daß es eine abgespeckte Teilmenge von SGML darstellt. Die Standard Generalized Markup Language (SGML) wurde 1969 aus der Generalized Markup Language (GML), einem IBM-Produkt, entwickelt. Sie ist seit 1986 ein ISO-Standard (ISO 8879). Aufgrund ihrer großen Komplexität und Variabilität wurde sie vor allem im Bereich der technischen Dokumentation bzw. im Prepress-Bereich in Verlagen angewandt. Die Komplexität der SGML-Sprache und der Stylekomponente DSSSL[2] ist der Grund dafür, daß die Werkzeuge, besonders die Browser, nicht kostenfrei zur Verfügung gestellt werden. Daher konnte sich SGML bisher nicht als Web-Publishing-Strategie durchsetzen.

Unterschiede zu HTML

Mit der Definition von HTML, welches eine dem SGML-Standard konforme Dokumenttypdefinition ist, wurde eine Möglichkeit geschaffen, durch relativ einfaches Markup Dokumente auszuzeichnen und diese über das WWW zur Verfügung zu stellen. Die Entwicklung von freier Software, wie Browsern und Autorensystemen, machten den Erfolg aus. Aktuelle Entwicklungen, vor allem aus dem e-Commerce-Bereich stellen jedoch neue Anforderungen an Webtechnologien, denen HTML nicht gerecht werden kann. Durch eine Vielzahl von inoffiziellen Weiterentwicklungen (diejenigen, die nicht vom World Wide Web Consortium[3] genormt wurden) versuchten Browser-Verkäufer, Marktanteile zu gewinnen, indem sie HTML mit inkompatiblen Erweiterungen ausstatteten, z.B. dem <BLINK>-Element, oder dem <CENTER>-Element. Somit wurde das Layout wieder ein Teil der Auszeichnung in HTML. Das führte natürlich dazu, daß die Interoperabilität des WWW eingeschränkt und Insellösungen geschaffen wurden, was jedoch der Philosophie einer weltweiten Standardisierung widerspricht. Aus diesem Grunde wurden vom W3C u.a. beschlossen, für die Formatierung von Dokumenten auf Style Sheets zurückzugreifen, eine Idee, die aus der SGML-Technologie stammt. Damit wurde die strikte Trennung von Text und Markup propagiert. Weiterhin wurde ein Mechanismus (Dynamic HTML: DHTML) definiert, um HTML mit Abstraktionen zu versehen.

Da diese Bemühungen, HTML wieder webfähig zu machen, jedoch auf die Dauer nicht haltbar sind, wurden 1996 erste Ansätze der Entwicklung von XML als einer Untermenge von SGML, die dessen Hauptvorteile nutzt und leichter über das Web zu transportieren ist, diskutiert. XML liegt in einer Empfehlung des W3C vor, die Spezifikation XML1.0 wurde vom W3C im Februar 1998 verabschiedet. Die aktuelle Definition von Januar 1999 befaßt sich mit der Definition der Namespaces, siehe Seiten des W3C: *http://www.w3.org/XML/*. Es gibt ebenso einen Zusatz zur ISO8879, das „technical Korregendum“ vom 4. Dezember 1997, indem Web-SGML als eine Erweiterung zu SGML definiert wird. Unter Web-SGML wird XML verstanden. XML bietet Vorteile gegenüber SGML. Damit erhöht sich die Wahrscheinlichkeit einer breiteren Verfügbarkeit von kostengünstigen Tools. Da XML ebenso eine breite Unterstützung in der Industrie findet, z.B. bei der Microsoft Corporation und bei Sun Microsystems, Inc., wird es, so hofft man, sowohl im Windows-Bereich als auch in der UNIX-Welt eine große Rolle spielen, vgl. [3], Seite 8.

Aus heutiger Sicht besitzt XML gegenüber HTML folgende Vorteile:

- In HTML steht nur ein begrenzter Satz an Befehlen zur Verfügung. HTML dient vor allem der Beschreibung der optischen Präsentation von Dokumenten. XML bietet dagegen die Möglichkeit, flexibel eigene Tags zu benutzen und die Dokumente somit mit einer feineren Struktur zu versehen, die die Darstellung der Dokumente erleichtert.
- Es vereinfacht das Publizieren von Datenbanken und den Austausch von Datenbankinformationen. Datenbanken nutzen meist einfache Dateiformate, um die Informationen auszutauschen, z.B. das Schema: ein Datensatz pro Zeile und ein Trennsymbol zwischen den einzelnen Feldern. Objektorientierte Datenbanken haben eine kompliziertere Struktur, die mit einfachen Modellen, wie z.B. Tabellen in HTML, nicht mehr ausgedrückt werden kann. Objekte müssen interne Strukturen mit dazwischenliegenden Links be-

2 DSSSL heißt Document Style Semantics and Specification Language und stellt die Stilkomponente von SGML dar.

3 Im folgenden wird für World Wide Web Consortium stets die Abkürzung W3C verwendet.

sitzen. XML kann dies mittels Elementen und Attributen abbilden und so die Struktur der Datenbank erhalten. Das vereinfacht den Austausch bzw. die Wiederverwendung der Daten in anderen Datenbanken.

- Ein weiterer Vorteil besteht darin, daß sich durch spezielle Entwicklungen im XML-Umfeld, wie zum Beispiel RDF (Resource Description Framework) neue Standards, z.B. für die Beschreibung von Webinhalten durch Metadaten, durchsetzen. Dadurch sind gezieltere Suchanfragen und Auswertungen von Webinhalten möglich.
- XML ermöglicht die Integration von Daten verschiedener Quellen in einer einheitlichen logischen Sichtweise. Diese kann z.B. in Internetmarktplätzen genutzt werden, um die Darstellung der vielen Produktkataloge zu vereinheitlichen.
- Mit der Linking Komponente von XML ist es nunmehr auch möglich, mehrdirektionale Links abzubilden. In HTML dagegen hat der Nutzer nur die Möglichkeiten Links mit einem Ziel durch <A HREF= ...> anzugeben. Man kann sogar Linkhierarchien aufbauen und so z.B. den Vorgänger eines Links referenzieren.

Ziele von XML

XML wurde, wie dargestellt, mit dem Ziel ins Leben gerufen, die Interoperabilität zwischen den verschiedenen Webanwendungen zu erleichtern und die herstellerunabhängige Darstellung und Verarbeitung strukturierter Daten zu ermöglichen. Die Verarbeitung verschiedenster Dokumenttypen, wie z.B. herkömmlicher WWW-Dokumente oder wissenschaftlicher Publikationen, soll ebenso ermöglicht werden wie die Darstellung und Interpretation von Suchergebnissen oder Metainhalten des WWW oder die Verarbeitung und Publikation strukturierter Einträge aus Datenbanken, Einkaufsformularen und Bestellungen etc. Durch die Nutzung des neuen Dokumenttyps für Metadaten, RDF (Resource Description Framework), soll die Katalogisierung, Beschreibung und Verwaltung von Webinhalten besser gelingen. Er bildet den neuen Metadatenstandard im WWW. Das Publizieren von Datenbanken wird sich zum besonders gut genutzten Anwendungsgebiet von XML entwickeln, da die Unterstützung objektorientierter Konzepte wie die Darstellung der inneren Struktur von Objekten oder deren hierarchischer und Linkbeziehungen nun einfacher möglich ist. XML kann auch sehr gut als Austauschformat zwischen verschiedenen Datenbanken dienen. Die Definition spezieller DTDs für mathematische Formeln (MathML) oder chemische Strukturen (CML) vereinfacht die Nutzung für bestimmte Wissenschaftsbereiche.

Im allgemeinen versteht man unter XML die folgenden drei Hauptkomponenten, die Sprachdefinition XML, die Stylesprache Extensible Style Language (XSL) und die Linking Komponente, die Extensible Linking Language (Xlink). Letztere besteht aus den Komponenten Xlink für einfache Verweise und Xpointer für multidirektionale Links.

Das Verhältnis von SGML, XML, HTML und ihrer Komponenten

XML ist eine Teilmenge von SGML, das gleiche gilt für die dazugehörende Style-Komponente Extensible Style Language (XSL), die einen Teil der Fähigkeiten der Document Style Semantics and Specification Language (DSSSL) besitzt. HTML ist eine SGML-DTD und könnte unter bestimmten Bedingungen auch eine XML-DTD sein. Cascading Style Sheets (CSS) sind die Stylekomponente von HTML. Sie sind ebenso auf XML-Dokumente anwendbar.

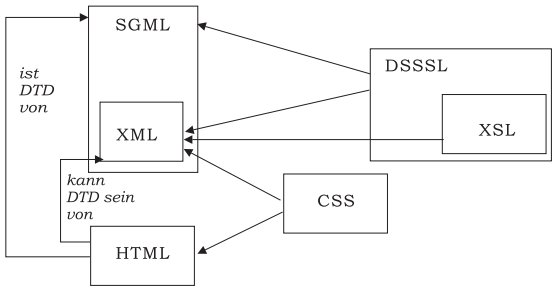

Abb. 1: Der Zusammenhang von SGML; XML und HTML, sowie der Stilkomponenten, aus [1].

Wie sehen XML-Dokumente aus?

Im folgenden wir vor allem auf die Darstellung und Verarbeitung von Text eingegangen.

Physikalische Struktur

Ein XML-Dokument ist definiert als eine Folge von Zeichen, die durch den *Unicode*-Zeichensatz definiert werden. Ein XML-Parser arbeitet das Dokument sequentiell durch. Dabei kann der Text nichtlinear in mehreren Stücken gespeichert werden. Ein Parser reorganisiert die lineare Struktur des Dokuments. Dabei berücksichtigt der Parser im Gegensatz zu SGML oder HTML z.B. auch die Groß- und Kleinschreibung der Tagnamen.

Ein XML-Dokument besteht aus einem Prolog und der Dokumentinstanz. Der Prolog ist optional und stellt Informationen darüber zur Verfügung, wie das Dokument physisch (wo sich seine Komponenten befinden) und logisch (wie sich die Elemente zusammenfügen) strukturiert ist (aus [2], Seite 456). Derartige Informationen sind z.B. die XML-Version oder der Dokumenttyp.

Die Dokumentinstanz beinhaltet die eigentlichen Daten in Form einer Elementhierarchie. Es können die in Tags eingeschlossenen Elemente (*Entities*) eines

Beispiel: einfacher Prolog eines XML-Dokuments:

<?xml version="1.0"? encoding="UTF-8">

Angabe der verwendeten XML-Version und der Kodierungsdeklaration. Letzere gibt an, welches Kodierungsschema verwendet wurde, z.B. 7-Bit-ASCII (UTF-8), einer Untermenge der Unicode-Codierung.

<!DOCTYPE notiz SYSTEM file://notiz.dtd" >

Angabe der benutzten Dokumenttypdeklaration, auf die das Dokument aufbaut. Diese Angabe ist nicht zwingend für alle XML-Dokumente.

Dokuments, Kommentare und CDATA[4]-Abschnitte vorkommen. Tags sind in spitze Klammer eingeschlossenen Bezeichner für Elementtypen, z.B. für den Elementtyp Autor: <autor> Hans Rudolf Muster </autor>. Grundsätzlich wird zwischen Elementen mit *leerem* und Elementen mit *nichtleerem* Inhalt unterschieden. Bei nichtleeren Elementen ist es sehr wichtig, daß es zu jedem Start-Tag < ... > auch ein sogenanntes End-Tag gibt, </...>. Dazwischen steht dann der Inhalt des Elements. Leere Elemente enden im Start-Tag nicht mit einer schließenden spitzen Klammern, sondern mit /> und sehen z.B. so aus: . Beide Elementarten können Attribute besitzen, so ist im letzten Beispiel src ein Attribut, welches mit dem Wert bild.gif belegt wurde.

Entities enthalten entweder ein oder mehrere Zeichen und besitzen einen Namen. Will man im Dokument ein vorher definiertes Entity nutzen, so muß man es referenzieren. Beim Parsen wird die Referenz dann durch den Inhalt, d.h. das Entity, ersetzt. In XML werden sogar externe Entities zugelassen. Darunter können sowohl Dateien als auch Objekte einer Datenbank verstanden werden. Entities können sogar quer über das Internet verteilt sein.

Beispiele:

Ein als Abkürzung verwendetes Entity:
<!ENTITY sd "Susanne Dobratz, Rechenzentrum">

Ein externes Entity:
<!ENTITY intro chapter SYSTEM
http://www.xyz.org/intro.xml>

Eine Referenz auf ein Entity:
... <addressee> &sd; Humboldt-University, Berlin
</addressee> ...

Es gibt in XML fünf wichtige vordefinierte Entities:

&amp für &, &lt für <, &gt für >, &apos für ' und &quot für ".

4 Darauf wird an späterer Stelle noch eingegangen

5 siehe Abschnitt Wohlgeformt und Gültig in diesem Artikel.

Diese Entities sind spezielle Zeichen, die sonst in einem XML-Dokument nicht direkt zu verwenden sind, da sie in XML eine eigene Bedeutung haben.

Kommentare werden in XML mit folgender Syntax eingegeben: <!-- Dies ist ein Kommentar --!>.

Normale Entities werden vor der Präsentation oder Verarbeitung auf ihre Wohlgeformheit[5] durch den Parser überprüft. Es sind aber auch Abschnitte in XML-Dokumenten zulässig, deren Inhalte (ganze Zeichenketten) ungeprüft durch den Parser gelangen. Diese heißen dann CDATA-Abschnitte. CDATA steht für Character Data (*Zeichendaten*). So können z.B. JAVA-Code oder andere Programmieranweisungen in CDATA-Abschnitten untergebracht werden.

Beispiel:

JAVA-Code in einem CDATA -Abschnitt (aus [2], Seite 453):

```
<![CDATA [
    if (foo.getContentLength() < 0 && input =
            foo.getInputStream())
    open = true;
]]>
```

Processing Instruktions (PI) bilden einen weiteren Bestandteil von XML-Dokumenten. Sie geben Anweisungen, wie ein Dokument zu verarbeiten ist und werden in <? und ?> eingeschlossen.

Beispiel:

für Processing Instructions (aus [1], Seite 70):

<?apache include file="Fusszeile.html" ?>
<?behme-mintert manuscript-status="draft" ?>

Logische Struktur

XML-Dokumente besitzen eine hierarchische, d.h. baumartige Struktur. Damit gibt es Elemente, die anderen Elementen untergeordnet sind und umgekehrt. In der folgenden Abbildung ist der hierarchische Aufbau

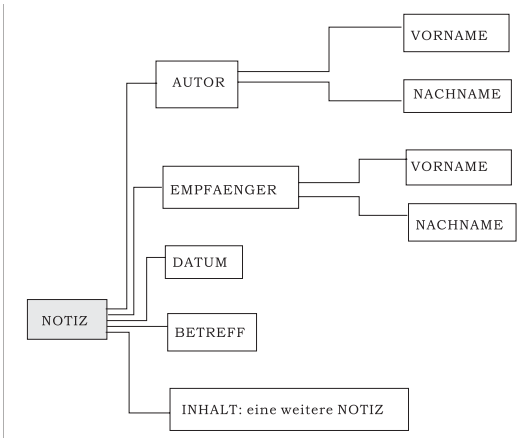

Abb. 2: Logischer Aufbau des Notizbeispiels

des Dokumenttyps Notiz dargestellt. Es muß in einem XML-Dokument immer ein sogenanntes Wurzel-Element existieren, das allen anderen Elementen übergeordnet ist.

Wohlgeformt und Gültig

Grundsätzlich spricht man von zwei Möglichkeiten, XML- Dokumente zu erstellen: wohlgeformte Dokumente und gültige Dokumente.

a) Wohlgeformte XML-Dokumente (wellformed documents)
Ein textuelles Objekt ist ein wohlgeformtes XML -Dokument, wenn es eine logische Struktur besitzt. Folgendes Dokument ist z.B. wohlgeformt:

```
<doc>
<title>  xyz  </title
<author >abc </author
<inhalt> jwhef </inhalt>
</doc>
```

Auffallend ist, daß die Schachtelung der Elemente hierarchisch erfolgt und daß es zu jedem Tag (in spitze Klammern eingeschlossener Bezeichner) <tagname> auch ein End-Tag </tagname>geben muß. Folgendes Dokument ist nicht wohlgeformt, da das Tag <autor> geöffnet wird, bevor das tag <title> geschlossen wurde:

```
<doc>
<title> nsdbfnvb
<autor>
</title>
</autor>
<inhalt> ,hcnfdv </inhalt
</doc>.
```

b) Gültige XML-Dokumente (valid documents)
Das bedeutet, daß es eine Dokumenttypdefinition (DTD) gibt, anhand derer das Dokument validiert werden kann.

Ist ein Dokument zu einer DTD konform, dann ist es gültig, wobei naturgemäß gültige XML-Dokumente auch immer wohlgeformt sind.

Wie kann man XML-Dokumente erstellen?

Da XML eine Teilmenge von SGML bildet, kann grundsätzlich jedes SGML-Werkzeug zur Erstellung von XML-Dokumenten benutzt werden. Die einfachste Möglichkeit besteht darin, einen einfachen Texteditor, der auf ASCII-Basis arbeitet, zu nutzen (siehe folgendes Beispiel).

Die Nutzung einfacher XML-Editoren, wie z.B. des XML-Notepads unter Windows95 bzw. 98, ist ebenso möglich, wie die Nutzung komplexer Publishing-Programme mit XML-Unterstützung, wie z.B. Frame-

Bsp. eines Notiz-(buch) mit zugehörigem CSS-File:

```
/*notiz1.xml*/
<?xml version="1.0" encoding="iso-8859-1"? >
<?xml:stylesheet  type="text/css" href="notiz1.css"?>
<NOTIZ >
  <AUTOR >
    <VORNAME>Susanne</VORNAME>
    <NACHNAME>Dobratz</NACHNAME>
  </AUTOR >
  <EMPFAENGER>
    <VORNAME>Matthias</VORNAME>
    <NACHNAME>Schulz</NACHNAME>
  </EMPFAENGER>
<DATUM>23.06.1999</DATUM>
  <BETREFF>Treffen</BETREFF>
  <INHALT>Lieber Matthias, koennen wir uns
morgen um 13.00 Uhr treffen? Gruss, Susanne
</INHALT>
</NOTIZ>

/* notiz1.css */

NOTIZ { display : block;
            margin-left : 1cm;
            margin-right: 3cm
            font-family:times;
            font-size : 12pt;
            background : #0ddfff;}

DATUM { display : block;
            text-align : center;
            margin-top : 0.5em;
            margin-bottom : 0.5em; }

INHALT { display : block;
       text-align : center; }

AUTOR { display : block;
            color : blue;
      text-align : block;
      font-style : italic;
      margin-top : 0.5em;
      margin-bottom : 0.5em;
      margin-left : 20em;}

EMPFAENGER { display : block;
            color : green;
      text-align : block;
      font-style : italic;
      margin-top : 0.5em;
      margin-bottom : 0.5em;
      margin-right : 5em; }

BETREFF { display : block;
        color : darkblue;
        font-family : arial;
        font-size : 16pt;
        font-weight : bold; }
```

Maker+SGML. Bei letzterem funktioniert die XML-Unterstützung allerdings im Moment nur richtig in eine Richtung, dem Export nach XML. Das Importieren von XML-Dokumenten funktioniert noch nicht richtig. Adobe bietet mit seinem neuen Programm GoLive4 die Möglichkeit XML-Daten zu öffnen und, siehe [5], sie rudimentär zu bearbeiten. Quark bietet mit seinem neuen Programm Troika ebenso ein System zur XML-Unterstützung an. Damit lassen sich XML-Daten aus XPress exportieren, aus [5]. Auch andere Textsatzsysteme sollen eine XML-Unterstützung bekommen. IBM bietet zum Beispiel eine TeXML-DTD mit zugehörigem Parser an. Damit ist es möglich, in XML TeX-Dokumente[6] zu schreiben, diese dann nach TeX zu konvertieren und mit gängigen TeX-Tools weiter zu verarbeiten. Es gibt weiterhin einen XML-Editor-Builder, der aus einer DTD einen XML-Editor generiert. Alle diese Applikationen sind jedoch Java-Programme. Das heißt, auf dem Rechner muß JAVA installiert sein, damit diese Werkzeuge nutzbar sind

Leider kann zur Zeit nur einer der gängigen WWW-Browser die Präsentation von XML-Dokumenten realisieren. Microsofts Internet Explorer 5.0 zeigt XML-Dateien entweder basierend auf Cascading Style Sheets oder XSL-Formatierungen an. Gibt es zu einem XML-Dokument keine der beiden Stildateien, wird das Dokument in seiner Baumstruktur angezeigt, in der Elemente beliebig ein- oder ausgeblendet werden können. Aber auch hier versucht Microsoft wieder, eigene Elemente in die XML bzw. XSL-Implementation hineinzubringen, die weitab des vom W3C definierten Standards liegen, vgl. [4]. So wird z.B. ein Namespace für HTML dadurch definiert, daß es als festes Prefix vor eine XML-Datei geschaltet werden kann. Damit ist es möglich, HTML-Elemente in XML-Seiten zu benutzen. Diese Definition widerspricht jedoch dem Standard. So sind derartige Seiten dann wieder nur „designed for Microsoft Internet Explorer X.X“. Der Netscape-Browser kann zum momentanen Zeitpunkt noch keine XML-Dokumente interpretieren. IBM bietet einen XML Viewer, ein Teil des Xplorer, auf Java-Basis an.

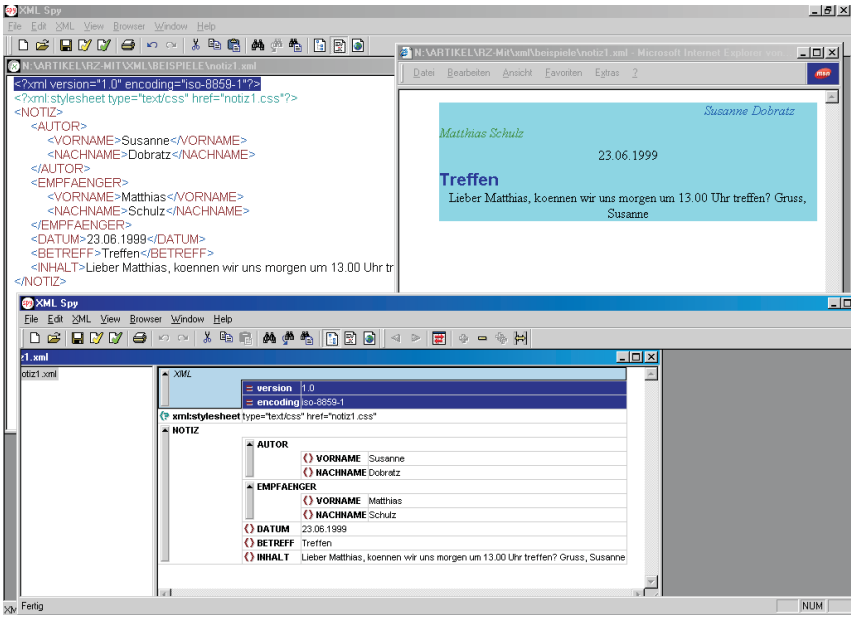

Abb. 3: Ansicht der Notiz im XML Spy 2.5: im Source und als Baumstruktur sowie im Internet Explorer 5.0 unter Nutzung des obigen CSS-Files.

Links to XML Sites

Seiten des W3C zu XML: http://www.w3.org/XML/

Robin Cover's bibliography and background information on SGML, HyTime, and DSSSL, since 1994: http://www.oasis-open.org/cover/

Microsoft XML-Tutorial: http://msdn.microsoft.com/xml/general/self-describing.asp

Alphaworksseite der IBM: http://www.alphaworks.ibm.com/

XML DeveloperSeiten von SUN: http://developer.java.sun.com/developer/technicalArticles/Networking/XMLAndJava/index.html

Literatur

[1] Henning Behme, Stefan Mintert: *XML in der Praxis*, Bonn, Addison-Wesley-Longman, 1998, ISBN: 3-8273-1330-9

[2] Charles F. Goldfarb, Paul Prescott: *XML Handbuch*, München, London: Prentice Hall, 1999, ISBN: 3-8272-9575-0

[3] David A. Patrick, XML in der Praxis – Unternehmensübergreifende Vorteile durch Enterprise Content Management – , in: *nfd 50*,1999, Seiten 5-12

[4] Rainold Menge: XML an allen Ecken, in *c't 8/1999*, Seite 34f

[5] Michael Kerbe: Quo Vadis, XML?, in: *MACUP 8/1999*, Seiten 118-120

Susanne Dobratz
susanne.dobratz@rz.hu-berlin.de

6 An dieser Stelle ist von TeX und nicht von LaTeX die Rede!