1 Texttechnologie

Georg Rehm

Die Texttechnologie stellt ein noch junges Forschungsfeld dar, das die linguistisch motivierte Informationsanreicherung und Verarbeitung digital verfügbarer Texte mit standardisierten Auszeichnungssprachen fokussiert. Eine zielgerichtete Definition ist aufgrund der zahlreichen, in konkreten Anwendungen potentiell beteiligten Disziplinen – u. a. Computer- und Korpuslinguistik, Text- und Hypertext-Theorie, Text-Mining – nicht ohne weiteres möglich, d. h. die Texttechnologie umfasst nicht eine eindeutig bestimmbare Menge aufeinander aufbauender Methoden oder Theorien, sie ist vielmehr „wissenschaftlich begründete Praxis“ (*Lobin und Lemnitzer* 2003a, S. 1). Als ‘kleinster gemeinsamer Nenner’ wird in allen texttechnologischen Anwendungen die Metasprache XML (Extensible Markup Language, *Bray et al.* 2000) eingesetzt, die die Definition beliebiger Auszeichnungssprachen (die auch als *Markup-Sprachen* oder *XML-Anwendungen* bezeichnet werden) erlaubt. XML ist also eine formale Sprache zur Spezifizierung konkreter Markup-Sprachen, die wiederum zur Auszeichnung (auch: *Annotation*) arbiträrer Informationseinheiten in textuellen Daten eingesetzt werden können. In computerlinguistischen Anwendungen geht es hierbei u. a. um die Auszeichnung linguistischer Informationen in Texten, die Verwendung von XML als Datenaustauschformat zur einheitlichen Repräsentation von Ressourcen und den Einsatz *einer* Datengrundlage in *unterschiedlichen* Applikationskontexten.

Die Wurzeln der Texttechnologie liegen im Bereich der medien- und plattformunabhängigen Textauszeichnung, die erstmals mit der Standard Generalized Markup Language (SGML, *ISO 8879* 1986) eine breite Verwendung gefunden hat. Das zentrale Merkmal der SGML- bzw. XML-basierten Informationsmodellierung (*Lobin* 2000) ist die strikte Trennung von Form und Struktur durch die Einführung einer Abstraktionsebene, in der ein Textfragment von einem deklarativen Etikett (etwa `ueberschrift`, `paragraph` oder `beispielsatz`) umschlossen wird, ohne dabei jedoch Textsatz- oder Layout-Anweisungen zu kodieren. Im Vordergrund steht also nicht die typographische Gestaltung einer Wortfolge, sondern die eindeutige Markierung ihrer logischen Funktion bzw. ihrer Semantik innerhalb eines spezifischen Dokumenttyps. Die Auszeichnungselemente – häufig schlicht *Elemente* oder *Tags* genannt – einer Markup-Sprache sind nicht in beliebiger Form kombinierbar: Eine *Dokumentgrammatik* legt explizit die *hierarchischen Kombinationsmöglichkeiten* hinsichtlich der Strukturierung von Elementen fest, weshalb die in einem annotierten Dokument enthaltenen Tags im graphentheoretischen Sinn einen Baum aufspannen (vgl. Abb. 1). Die Aufbereitung eines XML-annotierten Textes (auch: *Dokumentinstanz*) zu einem formatierten und publizierbaren Dokument erfolgt mit XSL/XSLT (Extensible Stylesheet Language, XSL Transformations, *Clark* 1999). Die Auslagerung der Abbildung einzelner Tags auf korrespondierende Layout-Anweisungen legte den Grundstein für das *Cross Media* bzw. *Single Source Publishing*. Hierunter versteht man die Möglichkeit, aus ein- und derselben annotierten Textquelle mit

Hilfe unterschiedlicher *Style Sheets* z. B. eine für den Einsatz im WWW aufbereitete HTML-Version und eine für den Druck optimierte PDF-Version generieren zu können (*Möhr und Schmidt* 1999, *Kreulich* 2003).

Abschnitt 1.1 thematisiert zunächst HTML, die *Lingua Franca* des World Wide Web, woraufhin Abschnitt 1.2 auf XML eingeht. Abschnitt 1.3 stellt unterschiedliche Verarbeitungsmethoden von XML-Instanzen vor, woraufhin Abschnitt 1.3.4 auf Standards eingeht, die XML flankieren und zusätzliche Funktionalität bereit stellen. Der sehr komplexe Standard SGML wurde mittlerweile fast vollständig von XML verdrängt, das eine Teilmenge von WebSGML (*ISO 8879 TC2* 1998) darstellt, wobei die Kernmerkmale von SGML – die Möglichkeit der Definition beliebiger Markup-Sprachen zur hierarchischen Strukturierung arbiträrer Informationen – beibehalten wurden. Diejenigen Eigenschaften, die die Implementierung von SGML-Prozessoren und die Verarbeitung von Instanzen unnötig komplex werden lassen, wurden aus Gründen der Vereinfachung bei der Spezifizierung von XML nicht berücksichtigt.

1.1 HTML – Hypertext Markup Language

Webdokumente werden mit Hilfe der Hypertext Markup Language (*Raggett et al.* 1999) ausgezeichnet, die zugleich die bekannteste Auszeichnungssprache darstellt (Unterkapitel **??**, *Das World Wide Web*, thematisiert die Nutzung von HTML-Dokumenten in sprachtechnologischen Anwendungen). HTML 4.01 spezifiziert 93 verschiedene Elemente, so markiert z. B. das Tag `<p>` (*paragraph*) einen Absatz, und die Struktur einer Tabelle wird durch das Element `<table>` (bestehend aus *table rows*, `<tr>`, die wiederum `<td>`-Elemente, *table data cell*, enthalten) modelliert. Weitere Elemente erlauben z. B. die Auszeichnung von Überschriften, Listen und vor allem die Integration von Hyperlinks mittels des Tags `<a>` (*anchor*), wobei das Attribut `href` die URL des Dokuments enthält, auf das verwiesen wird, z. B. `<a href="http://www.uni-giessen.de">JLU Gießen</a>`. Ein wichtiger Aspekt betrifft die Mischung unterschiedlicher Auszeichnungsebenen, so werden Elemente für strukturelles (z. B. `<table>` oder `<h1>` für eine Überschrift erster Stufe), logisches (`<em>`, `<strong>` zur Markierung wichtiger bzw. sehr wichtiger Textteile), präsentationsorientiertes (`<i>`, `<b>` für Kursiv- bzw. Fettdruck), referentielles (`<a>`) und funktionales Markup (`<object>`, zur Einbettung externer Programme) definiert (*Walker* 1999).

Die Dokumenttyp-Definitionen (DTD), d. h. die regelbasierten, formalen Definitionen, die die Namen und das Zusammenspiel von Elementen und Attributen spezifizieren, wurden bis zu HTML 4.01 mit Hilfe von SGML definiert. Im Jahr 2000 wurde erstmals eine Neuformulierung von HTML auf der Grundlage von XML vorgenommen, um die formale Kompatibilität mit dem XML-Paradigma zu gewährleisten. XHTML 1.0 (*Pemberton* 2002) ist dabei nur der erste Schritt, denn mit *Modularization of XHTML* (*Altheim et al.* 2001) steht mittlerweile ein Inventar zur Verfügung, dass die Anwendung speziell angepasster XHTML-Vokabularien erlaubt, wie z. B. *XHTML Basic* (*Baker et al.* 2000), das für mobile Endgeräte gedacht ist, oder XHTML 1.1 (*Altheim und McCarron* 2001), das als Grundlage für zukünftige modularisierte Versionen von XHTML dienen soll.

1.2 XML – Extensible Markup Language

Bereits 1994 wurde angeregt (*Sperberg-McQueen und Goldstein* 1994), das fest vorgeschriebene, statische Inventar von HTML-Elementen durch einen flexibleren Mechanismus zu ergänzen, der die Definition beliebiger Auszeichnungssprachen und ihren Einsatz im Web-Umfeld erlaubt. Gerade die mangelnde Erweiterbarkeit von HTML war für das World Wide Web Consortium (W3C, `http://www.w3.org`), das Web-bezogene Standards konzipiert und verabschiedet, der Anlass, die Extensible Markup Language (XML, *Bray et al.* 2000) zu spezifizieren, die eben diese Explizierung arbiträrer Informationen ermöglicht, indem eine strikte Trennung von Inhalt und Struktur vorgenommen wird. Dieser Strukturaspekt bezieht sich dabei in vielen XML-basierten Markup-Sprachen – im Übrigen auch in den meisten XML-Einführungstexten – vornehmlich auf logische Texteinheiten, die auf der Makroebene anzusiedeln sind, z. B. *Tabelle*, *Überschrift*, *Absatz* oder *Liste* oder einzelne Binnenstrukturen der Mikroebene, etwa die Aufspaltung einer Postanschrift in *Empfänger* (bestehend aus *Vorname*, *Nachname*) und *Anschrift* (*Straße*, *Hausnummer*, *Postleitzahl*, *Stadt*, *Land*). Tatsächlich sind mit XML-basierten Markup-Sprachen jedoch *beliebige Strukturen* annotierbar, im linguistischen Bereich z. B. die rhetorische Struktur eines Textes auf der Grundlage der Rhetorical Structure Theory (RST, vgl. etwa *Lobin* 1999a und *Rehm* 1999 sowie Unterkapitel **??**, *Text, Diskurs und Dialog*) oder die Phrasenstruktur einzelner Sätze. Abb. 1 zeigt mit einer XML-Dokumentinstanz ein derartiges Beispiel (nach *Witt* 1999) sowie die von den XML-Elementen aufgespannte Baumstruktur, die der syntaktischen Struktur des Satzes entspricht. Die zugehörige Dokumentgrammatik (DTD) folgt einer speziellen Syntax, die im XML-Standard festgelegt ist, und enthält im Wesentlichen die Namen von Elementen und Attributen sowie die Kombinationsmöglichkeiten von Elementen, die durch sog. *Inhaltsmodelle* spezifiziert werden.

```
<!ELEMENT s  (np, vp)>        <!ELEMENT v   (#PCDATA)>
<!ELEMENT np (det, n)>        <!ELEMENT n   (#PCDATA)>
<!ELEMENT vp (v, np*)>        <!ELEMENT det (#PCDATA)>

<!ATTLIST np kasus (nom|akk|dat|gen)   #REQUIRED>
<!ATTLIST v  agr   (1s|2s|3s|1p|2p|3p) #REQUIRED>
```

Die ersten drei Zeilen der DTD deklarieren, eingeleitet durch das Schlüsselwort `ELEMENT`, sechs Auszeichnungselemente sowie die jeweiligen Inhaltsmodelle; hierbei ist zu beachten, dass `s` das Wurzelelement der DTD ist, also das äußerste Element einer Instanz sein muss. Das Element `s` muss laut Inhaltsmodell in einer Dokumentinstanz zunächst das Element `np` gefolgt von dem Element `vp` enthalten. Diese Sequenz wird durch den Konnektor `,` erzwungen; als weiterer Konnektor ist das Zeichen `|` erlaubt, das eine *entweder oder*-Beziehung zwischen Elementen spezifiziert. Die Anzahl der Vorkommen einzelner Elemente wird – ähnlich wie in regulären Ausdrücken – durch Okkurrenzindikatoren festgelegt. Im Inhaltsmodell von `s` befinden sich keine Okkurrenzindikatoren, weshalb in einer Instanz, die nach dieser DTD annotiert wird, jeweils genau ein `np`- und ein

```
<?xml version="1.0" encoding="UTF-8"?>
<s>
    <np kasus="nom">
        <det>Der</det>
        <n>Mann</n>
    </np>
    <vp>
        <v agr="3s">tritt</v>
        <np kasus="akk">
            <det>den</det>
            <n>Ball</n>
        </np>
    </vp>
</s>
```

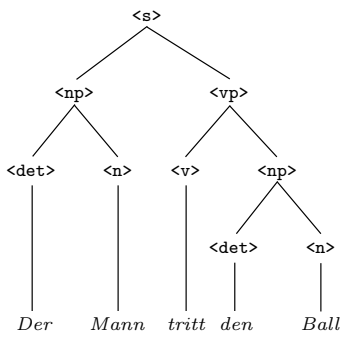

Abbildung 1: Eine XML-Dokumentinstanz am Beispiel der Annotation der Phrasenstruktur des Satzes „Der Mann tritt den Ball“

`vp`-Element enthalten sein müssen. In der Deklaration des Elements `vp` hingegen ist durch `*` markiert, dass dem Tag `v` 0..n `np`-Elemente folgen dürfen (zur groben Modellierung intransitiver, transitiver und ditransitiver Verben). Die beiden weiteren Okkurrenzindikatoren sind `?` und `+`, mit denen 0..1 bzw. 1..n Vorkommen spezifiziert werden können. Mit Hilfe der Klammerung einzelner Teile eines Inhaltsmodells sowie der Angabe von Okkurrenzindikatoren an diesen Untermodellen sind komplexe Inhaltsmodelle realisierbar.

Die Elemente `s`, `np` und `vp` werden auch als *Containerelemente* bezeichnet, weil sie weitere Elemente aufnehmen, also noch keine konkreten Daten enthalten. Die drei *Datenelemente* `v`, `n` und `det` enthalten hingegen die Angabe des Schlüsselworts `#PCDATA` (*parsed character data*), d. h. sie dürfen nur konkrete Inhalte und keine Markup-Elemente enthalten. Mit Hilfe von Attributen können innerhalb eines Elements zusätzliche Informationen hinterlegt werden – hierbei gilt die Faustregel, dass sich Attribute nur auf Metadaten, d. h. Informationen *über* die annotierten Daten, beziehen sollten (*Maler und Andaloussi* 1996). *Schmidt* 2003 diskutiert die wesentlichen Aspekte und Standards bei der Auszeichnung von Metadaten im World Wide Web. Neben RDF/RDFS (vgl. Abschnitt 1.3.4) geht sie auf HTML und Dublin Core ein (`http://purl.org/dc/`). In Attributlistendeklarationen (`ATTLIST`) wird zunächst festgelegt, auf welches Element sich ein Attribut bezieht. In der Beispiel-DTD werden die beiden Attribute `kasus` und `agr` definiert, die für `np` bzw. `v` gelten. Es existieren unterschiedliche Typen von Attributen, die in einer Instanz u. a. die Eingabe eines beliebigen Textes als Wert erlauben. Das Beispiel zeigt jedoch den Aufzählungstyp, bei dem innerhalb der Deklaration die erlaubten Werte spezifiziert werden. Dabei gibt `#REQUIRED` an, dass ein Attribut obligatorisch ist (Optionalität wird durch `#IMPLIED` ausgedrückt). Neben den im Beispiel aufgeführten Syntaxelementen einer DTD existieren noch Entitäten, die als Platzhalter für textuelle Inhalte eingesetzt werden können sowie verschiedene Ausnahmen, die bei der DTD-Erstellung beachtet werden müssen, z. B. sind mehrdeutige Inhaltsmodelle wie `((a, c) | (a, d))` verboten, können jedoch meist umformuliert werden: `(a, (c | d))`.

1.2.1 Schemasprachen

Das syntaktische Inventar zur Spezifizierung von DTDs ist relativ eingeschränkt, weshalb mit XML Schema (*Fallside et al.* 2001) eine sehr komplexe Spezifikation standardisiert wurde, die präzisere Restriktionen in Instanzen ermöglicht. XML Schema gestattet z. B. die freie Definition von Datentypen, die Typisierung von Elementen sowie die Möglichkeit, die Anzahl der Vorkommen eines Elements innerhalb eines Inhaltsmodells genauer spezifizieren zu können als dies durch die Angabe von Okkurrenzindikatoren in einer DTD geschehen kann. Ein weiterer Vorteil von XML Schema und parallel entwickelten, alternativen Schemasprachen wie Relax NG oder Schematron besteht darin, dass Schemabeschreibungen ebenfalls in einer XML-Notation kodiert werden, d. h. die proprietäre Syntax, die innerhalb von DTDs benutzt wird, wurde aufgegeben, so dass Schemabeschreibungen – ebenso wie XML-Instanzen – mit beliebigen XML-Werkzeugen verarbeitet werden können.

1.2.2 Formale Eigenschaften von XML

Die Syntax einer formalen bzw. natürlichen Sprache wird durch ein Startsymbol, Produktionsregeln, einem terminalen sowie einem non-terminalen Vokabular definiert (vgl. Unterkapitel **??**, *Automatentheorie und Formale Sprachen*). Analog zu dieser Terminologie kann das Wurzelelement einer DTD als Startsymbol und die einzelnen Elementdeklarationen als Syntaxregeln bezeichnet werden; Containerelementen stellen das non-terminale Vokabular dar, Datenelemente umfassen die terminalen Symbole. Diese Analogie zu kontextfreien Grammatiken lässt sich jedoch nur in terminologischen Aspekten finden, denn DTDs und die verschiedenen Schema-Sprachen erzeugen keine linearen Ketten, sondern hierarchisch angeordnete Baumstrukturen, weshalb sie *reguläre Baumgrammatiken* genannt werden (*Lobin* 2003, *Mönnich und Morawietz* 2003). DTDs erlauben hierbei nicht die Benutzung abstrakter Symbole in Regeln, weshalb sie als *lokale Baumgrammatiken* bezeichnet werden.

1.3 Verarbeitung XML-annotierter Daten

Der wichtigste Aspekt im Umgang mit XML-Instanzen betrifft deren Verarbeitung mit XML-Parsern (z. B. expat, Xerxes oder MS XML), die nach dem gleichen Prinzip arbeiten wie Parser für natürliche Sprachen: Die DTD fungiert als formale Grammatik, gegen die eine Dokumentinstanz überprüft werden kann. Falls dies fehlerfrei abläuft, nennt man eine Instanz *valide*. Durch das Parsing können zahlreiche Fehler aufgedeckt werden, z. B. unbekannte Element- oder Attributnamen bzw. -werte oder ungültige Elementschachtelungen. XML-Instanzen müssen jedoch nicht zwangsläufig einer DTD zu Grunde liegen, sie können von einem Parser auch isoliert verarbeitet werden, wobei dann lediglich überprüft werden kann, ob das Dokument der im Standard definierten Basissyntax entspricht (öffnende Elemente beginnen mit dem Zeichen `<` und enden mit `>`, Elementschachtelungen dürfen sich nicht überlappen etc.), weshalb man von der Überprüfung auf *Wohlgeformtheit* spricht.

1.3.1 Betrachtung und Transformation

Die Betrachtung von XML-Instanzen ist mittlerweile mit den verbreiteten Browsern und zahlreichen frei erhältlichen Werkzeugen möglich, so bieten etwa die aktuellen Versionen von Mozilla und des Internet Explorers Möglichkeiten der Visualisierung der Baumstruktur sowie der integrierten XSLT-Transformation einer Instanz nach XHTML oder der Zuordnung eines Cascading Style Sheets (CSS), um einzelne Elemente kontextabhängig formatieren zu können.

XSLT-Transformationen beruhen auf dem Prinzip der regelbasierten Überführung eines XML-Dokuments z. B. in ein Präsentationsformat, zur Umstrukturierung oder, dies stellt zugleich die häufigste Anwendung dar, zur Konvertierung einer Instanz der DTD x in eine Instanz der DTD y (z. B. DocBook → XHTML, vgl. `http://www.oasis-open.org/docbook/` sowie `http://www.docbook.org`). Insbesondere für die Printausgabe wird XSL-FO (Formatting Objects) benutzt, das gemeinsam mit XSLT (*Clark* 1999) und XPath (*Clark und DeRose* 1999) den Standard XSL (Extensible Stylesheet Language, *Adler et al.* 2001) bildet. XPath übernimmt dabei die Rolle eines Hilfsstandards, der die Navigation in einem Dokument, genauer gesagt, in dessen Baumrepräsentation, und die Auswahl abstrakter Knoten (Elemente, Attribute, Text) ermöglicht. Mit Hilfe von XPath-Ausdrücken ist es in einem XSLT-Stylesheet möglich, für einzelne Elemente einer Dokumentinstanz korrespondierende Templates (Transformationsregeln) zu implementieren. XSLT wiederum ist als eine Markup-Sprache realisiert, d. h. XSLT-Stylesheets sind zugleich XML-Instanzen und können somit – wie Schemabeschreibungen – selbst zum Gegenstand XML-basierter Verarbeitungsprozesse werden. Zur Anwendung eines Stylesheets ist ein XML-Parser nicht ausreichend, da dieser lediglich Auskunft über die Validität bzw. Wohlgeformtheit einer Instanz geben kann, weshalb ein dezidierter XSLT-Prozessor wie z. B. Xalan, Saxon oder Sablotron benötigt wird.

1.3.2 Datenstrom- vs. baumbasierte Verarbeitung

Neben XSLT werden häufig zwei weitere Paradigmen eingesetzt, um in Programmiersprachen wie Java oder Perl die Verarbeitung XML-annotierter Daten zu ermöglichen: SAX (Simple API for XML, `http://www.saxproject.org`) erlaubt die Verarbeitung als Datenstrom, wobei das Auftreten eines öffnenden oder schließenden Tags oder von Fließtext spezielle Ereignisse auslösen (*event-based processing*), die von Funktionen verarbeitet werden können. Da die Instanz nicht vollständig in den Speicher eingelesen werden muss, eignet sich SAX insbesondere zur Verarbeitung extrem großer Datenmengen sowie für einfache Filterapplikationen. Im Gegensatz dazu erzeugt ein DOM-Prozessor (Document Object Model, *Hors et al.* 2000) eine interne Baumrepräsentation (*tree-based processing*). Der DOM-Standard definiert zahlreiche Methoden zur Navigation innerhalb des Baums und zur Manipulation von Knoten. Dies ermöglicht u. a. die Implementierung rekursiver Funktionen, weshalb DOM meist dann benutzt wird, wenn die Funktionalität von XSLT nicht mehr ausreicht (z. B. zur Aktualisierung einer Instanz mit Informationen, die aus einer Datenbank eingelesen werden).

1.3.3 Datenhaltung von XML-Instanzen

Zur Datenhaltung existieren verschiedene Möglichkeiten, von denen die häufigste die Pflege der Instanzen innerhalb des Dateisystems darstellt. Alternativ können relationale Datenbanken eingesetzt werden, wobei XML-Dokumente entweder vollständig in einer Tabelle gespeichert oder dynamisch aus mehreren Datensätzen ausgelesen, in eine Dokumentschablone integriert und anschließend exportiert werden. Die interessanteste Möglichkeit betrifft den Einsatz nativer XML-Datenbanken. Dieser neuartige Datenbanktyp, der eben nicht auf dem relationalen oder objektorientierten Paradigma basiert, speichert Instanzen in internen Datenstrukturen ab und unterstützt neben der Validierung zahlreiche Zugriffsmöglichkeiten, die üblicherweise auf XPath und XML Query basieren.

1.3.4 Flankierende XML-Standards

Um XML gruppieren sich zahlreiche, teils verabschiedete, teils noch in der Entwurfsphase befindliche W3C-Standards, die Schwerpunkte in einzelnen Anwendungsbereichen setzen. Namespaces erlauben z. B. die gleichzeitige Verwendung *mehrerer* Auszeichnungssprachen in *einer* XML-Instanz. Hierzu werden im Wurzelelement die korrespondierenden Namensräume sowie jeweils ein Präfix deklariert, das in öffnenden und schließenden Tags benutzt wird, um explizit ein spezielles Schema zu referenzieren (z. B. `<myns:doc>`). Die XML Linking Language (XLink) wurde zur Verknüpfung von Instanzen, aber auch zur Verbindung von Ressourcen im WWW entworfen. Die von XLink definierten Funktionen gehen weit über die einfachen Links hinaus, die HTML realisiert, so werden z. B. bidirektionale Links und Typisierungen ermöglicht. Mit SVG steht eine XML-basierte Markup-Sprache zur Verfügung, mit deren Hilfe Vektorgraphiken beschrieben werden können. SVG bietet insbesondere zur Visualisierung XML-annotierter Texte interessante Möglichkeiten, so sind XSLT-gesteuerte Transformationen von Textinstanzen in SVG-Instanzen möglich, die auch mit dynamischen Navigationselementen versehen werden können. Der Bereich der Web Services, die die Web-gestützte Kommunikation von Applikationen mittels SOAP (Simple Object Access Protocol) und WSDL (Web Services Description Language) spezifizieren, wird in *Wolff* 2003 aus computerlinguistischer Perspektive dargestellt.

In sprach- bzw. texttechnologischen Anwendungen werden häufig die Standards RDF (Resource Description Framework, *Lassila und Swick* 1999) und XML Topic Maps (XTM, *Pepper und Moore* 2001) zur Modellierung von semantischen Netzen und Ontologien eingesetzt (siehe Unterkapitel **??**, *Nichtsprachliches Wissen*). RDF, konzipiert zur Annotation von Metadaten, arbeitet mit den Objekttypen *Resources* (die Entität, über die eine Aussage getroffen wird), *Properties* (die spezifische Eigenschaft der zu beschreibenden Entität) und *Statements*. Ein Statement umfasst dabei eine Ressource (Subjekt), die Eigenschaft (Prädikat) sowie den annotierten Wert (Objekt). Häufig genannte Beispiele für Metadaten – Daten über Daten – sind der Autor einer Informationsressource, deren Titel oder die assoziierte Organisation. Zur Spe-

```
<rdf:RDF>
    <rdf:Description about="http://www.w3.org/Home/Lassila">
        <s:Creator>
            <rdf:Description about="http://www.w3.org/staffId/85740">
                <v:Name>Ora Lassila</v:Name>
                <v:Email>lassila@w3.org</v:Email>
            </rdf:Description>
        </s:Creator>
    </rdf:Description>
</rdf:RDF>
```

Abbildung 2: Eine RDF-Beschreibung in XML-Notation

zifizierung einer RDF-Beschreibung kann man ein Metadatum als natürlichsprachlichen Satz formulieren, z. B. „Ora Lassila ist der Autor der Ressource `http://www.w3.org/Home/Lassila`“. Subjekt, Prädikat und Objekt dieses Satzes entsprechen dabei nicht notwendigerweise den korrespondierenden Bestandteilen eines Statements, so wird in diesem Beispiel eine Aussage über die *Resource* `http://www.w3.org/Home/Lassila` (das Subjekt der RDF-Beschreibung) getroffen, wobei die *Property* (das Prädikat) `Autor` mit dem Wert `Ora Lassila` (das Objekt) markiert. Abb. 2 zeigt dieses Beispiel in erweiterter Form. RDF spezifiziert lediglich eine Syntax zur Beschreibung benannter Eigenschaften von Ressourcen sowie assoziierter Werte. Die eigentliche Mächtigkeit wird erst durch den derzeit in der Konzeptionierung befindlichen Standard RDF Schema (RDFS, *Brickley und Guha* 2003) erreicht. RDFS gestattet die Spezifizierung von RDF-Vokabularien durch die Definition typisierter Relationen und Eigenschaften, wobei auch Vererbungen von Eigenschaften modellierbar sind, so dass letztlich semantische Netze oder Ontologien beschrieben werden können. Diese Anwendung war die eigentliche Motivation des für SGML entwickelten Standards Topic Maps (*ISO/IEC 13250* 2000), der mit XML Topic Maps (XTM, *Pepper und Moore* 2001) auch in einer XML-Version vorliegt. In XTMs werden Knoten (*Topics*) mit Kanten (*Associations*) verbunden und u. U. mit einem Typensystem angereichert. Dieses abstrakte Wissen kann anschließend mit konkreten Instanzen (*Occurrences*) verbunden werden.

1.4 Texttechnologie und Computerlinguistik

Innerhalb sprachverarbeitender Anwendungen wird XML in verschiedenen Bereichen eingesetzt – die Annotation von Korpora ist das prominenteste Beispiel (vgl. Unterkapitel **??**, *Textkorpora*). Hierzu werden häufig die von der Text Encoding Initiative (TEI, `http://www.tei-c.org`, *Sperberg-McQueen und Burnard* 2002) entwickelten DTDs benutzt, die die Annotation unterschiedlichster Textsorten (Gedichte, Dramen, historische Materialien, Lexika) auf verschiedenen Ebenen erlauben: Zunächst kann man verschiedene Metadaten von Dokumenten erfassen. Die zweite Ebene beinhaltet die Auszeichnung textueller Einheiten wie *Band*, *Kapitel* oder *Abschnitt*. Auf der dritten Ebene werden mit *Sätzen* oder *Wörtern* Strukturen innerhalb von Abschnitten markiert. Die vierte Ebene

umfasst schließlich die Markierung *syntaktischer oder morphologischer Einheiten* (*Ide und Véronis* 1994, *Witt* 2003). Ein keinesfalls trivialer Aspekt betrifft die Zeichensatzkodierung: XML basiert auf Unicode, wodurch beliebige Alphabete repräsentierbar sind (*Sasaki und Witt* 2003). Generell konzentriert sich die Anwendung von XML in sprachverarbeitenden Systemen auf den Bereich der textuellen Datenbanken (*Lobin* 1999a) und eine Auswertung und Aufbereitung der Daten auf den angesprochenen Ebenen. Dabei geht es um die manuelle oder automatische Annotation von Texten (*McKelvie et al.* 1997, *Ule und Hinrichs* 2003, *Ule und Müller* 2003) und die Manipulation sowie die Verwendung des annotierten Materials in konkreten Anwendungsszenarien (*Möhr und Schmidt* 1999, *Lobin und Lemnitzer* 2003b, *Mehler und Lobin* 2003).

Die automatische Annotation von Daten kann in den unterschiedlichsten Anwendungsszenarien und prinzipiell mit beliebigen computerlinguistischen Methoden erfolgen (vgl. Kapitel **??**). Ein denkbares Szenario betrifft das in Abb. 1 gezeigte Beispiel der Annotation eines Satzes mit Phrasenstruktur-Elementen. Ein syntaktischer Parser für ein Fragment des Deutschen könnte derartige Annotationen aus der internen Syntaxrepräsentation des Eingabesatzes erzeugen, die daraufhin automatisch mit annotierten Testsätzen verglichen werden können (vgl. *Volk* 1998). In diesem speziellen Beispiel kodiert die DTD Syntaxregeln (z. B. VP $\rightarrow$ V NP *), was z. B. bei der Implementierung und Evaluation von Chunk-Parsern ausgenutzt werden kann (vgl. *Ule und Müller* 2003, siehe auch Unterkapitel **??**, *Syntax und Parsing*), die auf Eingabesätzen operieren, die aus Webseiten gewonnen wurden. Zu diesem Zweck muss zunächst ein Korpus aufgebaut werden (vgl. Unterkapitel **??**, *Das World Wide Web*), woraufhin die Dokumente tokenisiert und Satzgrenzen ermittelt werden müssen. Ein zentraler Aspekt texttechnologischer Verfahren, die XML-Instanzen maschinell erzeugen, gründet sich in deren Validierbarkeit gegen eine DTD. Beim Parsing einer automatisch erzeugten Instanz gegen die von der DTD erlaubten Strukturen resultieren fehlerhafte Annotationsalgorithmen unmittelbar in ungültigem Markup, was wiederum die Ausgabe einer Fehlermeldung des XML-Parsers bewirkt (*Rehm* 1999), die zur Evaluation des Systems benutzt oder auch automatisch ausgewertet werden kann, um dynamisch neue Annotationsregeln zu generieren.

1.5 Literaturhinweise

Die Spezifikationen von XML sowie den beteiligten Standards pflegt das W3C (`http://www.w3.org`). `http://www.edition-w3c.de` bietet deutsche Übersetzungen an, die – mit ergänzenden Kommentaren und Beispielen versehen – auch in Buchform erhältlich sind (*Mintert* 2002). Wichtige Ressourcen sind `http://xml.coverpages.org` und `http://www.xml.com`. Licht in den von zahlreichen Abkürzungen gezeichneten Dschungel der Text- und Web-Technologien bringen `http://www.xml-acronym-demystifier.org` und `http://wildesweb.com/glossary/`. Die Beiträge in *Lobin* 1999b und *Mehler und Lobin* 2003 stellen Verknüpfungen von texttechnologischen und computerlinguistischen Methoden dar. Mit *Lobin und Lemnitzer* 2003b liegt erstmals ein Band vor, der in die unterschiedlichen Facetten der Texttechnologie einführt.

Literatur

Adler, S., Berglund, A., Caruso, J., Deach, S., Graham, T., Grosso, P., Gutentag, E., Milowski, A., Parnell, S., Richman, J. und Zilles, S. (2001): „Extensible Stylesheet Language (XSL) 1.0“. Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/xsl/`.

Altheim, M., Boumphrey, F., Dooley, S., McCarron, S., Schnitzenbaumer, S. und Wugofski, T. (2001): „Modularization of XHTML“. Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/xhtml-modularization/`.

Altheim, M. und McCarron, S. (2001): „XHTML 1.1 – Module-based XHTML“. Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/xhtml11/`.

Baker, M., Ishikawa, M., Matsui, S., Stark, P., Wugofski, T. und Yamakami, T. (2000): „XHTML Basic“. Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/xhtml-basic/`.

Bray, T., Paoli, J., Sperberg-McQueen, C. M. und Maler, E. (2000): „Extensible Markup Language (XML) 1.0 (Second Edition)“. Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/2000/REC-xml-20001006`.

Brickley, D. und Guha, R. (2003): „RDF Vocabulary Description Language 1.0: RDF Schema“. Technische Spezifikation (Working Draft), World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/rdf-schema/`.

Clark, J. (1999): „XSL Transformations (Version 1.0)“. Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/xslt/`.

Clark, J. und DeRose, S. (1999): „XML Path Language (XPath) – Version 1.0“. Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/xslt/`.

Fallside, D. C., Thompson, H. S., Beech, D., Maloney, M., Mendelsohn, N., Biron, P. V. und Malhotra, A. (2001): „XML Schema“. Technische Spezifikation, World Wide Web Consortium. W3C Recommendation. Besteht aus Part 0 (Primer), Part 1 (Structures), Part 2 (Datatypes). Online verfügbar: `http://www.w3.org/XML/Schema`.

Hors, A. L., Hégaret, P. L., Wood, L., Nicol, G., Robie, J., Champion, M. und Byrne, S. (2000): „Document Object Model (DOM) Level 2 Core Specification“. Technische Spezifikation, World Wide Web Consortium.

Ide, N. und Véronis, J. (1994): „MULTEXT: Multilingual Text Tools and Corpora“. In: *COLING 94 – The 15th International Conference on Computational Linguistics*, Bd. 1, S. 588–592. Association for Computational Linguistics, Kyoto, Japan.

ISO 8879 (1986): „Information Processing – Text and Office Information Systems – Standard Generalized Markup Language“. Internationaler Standard, Genf, International Organization for Standardization.

ISO 8879 TC2 (1998): „ISO/IEC JTC1/SC34 N0029 – Document Description and Processing Languages – Technical Corrigendum 2 to ISO 8879:1986 (Annex K: WebSGML Adaptations; Annex L: Declaration for XML)“. Anhang/Revision eines

internationalen Standards, Genf, International Organization for Standardization. Online verfügbar: `http://www.sgmlsource.com/8879/n0029.htm`.

ISO/IEC 13250 (2000): „Information Technology – Document Description and Processing Languages – Topic Maps“. Internationaler Standard, Genf, International Organization for Standardization. Online verfügbar: `http://www.ornl.gov/sgml/wg4/`.

Kreulich, K. (2003): „Synthese logischer Texteinheiten im elektronischen Publizieren“. In: *Mehler und Lobin* (2003). Erscheint.

Lassila, O. und Swick, R. R. (1999): „Resource Description Framework (RDF). Model and Syntax Specification“. Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/REC-rdf-syntax/`.

Lobin, H. (1999a): „Intelligente Dokumente – Linguistische Repräsentation komplexer Inhalte für die hypermediale Wissensvermittlung“. In: *Lobin* (1999b), S. 155–178.

Lobin, H., Hrsg. (1999b): *Text im digitalen Medium – Linguistische Aspekte von Textdesign, Texttechnologie und Hypertext Engineering*. Wiesbaden: Westdeutscher Verlag.

Lobin, H. (2000): *Informationsmodellierung in XML und SGML*. Berlin, Heidelberg, New York etc.: Springer.

Lobin, H. (2003): „Textauszeichnung und Dokumentgrammatiken“. In: *Lobin und Lemnitzer* (2003b), S. 51–82. Erscheint.

Lobin, H. und Lemnitzer, L. (2003a): „Text(e) technologisch“. In: *Lobin und Lemnitzer* (2003b), S. 1–9. Erscheint.

Lobin, H. und Lemnitzer, L., Hrsg. (2003b): *Texttechnologie – Anwendungen und Perspektiven*. Tübingen: Stauffenburg. Erscheint.

Maler, E. und Andaloussi, J. E. (1996): *Developing SGML DTDs – From Text to Model to Markup*. Upper Saddle River: Prentice Hall.

McKelvie, D., Brew, C. und Thompson, H. (1997): „Using SGML as a Basis for Data-Intensive NLP“. In: *Proceedings of Applied Natural Language Processing (ANLP) 97*. Association for Computational Linguistics, Washington D. C. Online verfügbar: `http://www.ltg.hcrc.ed.ac.uk/~dmck/anlp97.ps`.

Mehler, A. und Lobin, H., Hrsg. (2003): *Automatische Textanalyse – Systeme und Methoden zur Annotation und Analyse natürlichsprachlicher Texte*. Wiesbaden: Westdeutscher Verlag. Erscheint.

Mintert, S., Hrsg. (2002): *XML & Co. Die W3C-Spezifikationen für Dokumenten- und Datenarchitektur*. Edition W3C.de. München, Boston, San Francisco etc.: Addison-Wesley.

Möhr, W. und Schmidt, I., Hrsg. (1999): *SGML und XML – Anwendungen und Perspektiven*. Berlin, Heidelberg, New York etc.: Springer.

Mönnich, U. und Morawietz, F. (2003): „Formale Grundlagen“. In: *Lobin und Lemnitzer* (2003b), S. 109–141. Erscheint.

Pemberton, S. (2002): „XHTML 1.0: The Extensible Hypertext Markup Language (Second Edition)“. Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/xhtml1/`.

Pepper, S. und Moore, G. (2001): „XML Topic Maps (XTM) 1.0“. Technische Spezifikation, TopicMaps.Org. Online verfügbar: `http://www.topicmaps.org/xtm/1.0/`.

Raggett, D., Hors, A. L. und Jacbos, I. (1999): „HTML 4.01 Specification“. Technische Spezifikation, World Wide Web Consortium. Online verfügbar: `http://www.w3.org/TR/html401/`.

Rehm, G. (1999): „Automatische Textannotation – Ein SGML- und DSSSL-basierter Ansatz zur angewandten Textlinguistik“. In: *Lobin* (1999b), S. 179–195.

Sasaki, F. und Witt, A. (2003): „Linguistische Korpora“. In: *Lobin und Lemnitzer* (2003b), S. 195–216. Erscheint.

Schmidt, I. (2003): „Modellierung von Metadaten“. In: *Lobin und Lemnitzer* (2003b), S. 143–164. Erscheint.

Sperberg-McQueen, C. M. und Burnard, L., Hrsg. (2002): *TEI P4: Guidelines for Electronic Text Encoding and Interchange*. Text Encoding Initiative Consortium; Humanities Computing Unit, University of Oxford.

Sperberg-McQueen, C. M. und Goldstein, R. F. (1994): „HTML to the Max – A Manifesto for Adding SGML Intelligence to the World-Wide Web“. In: *Proceedings of the Second International WWW Conference – Mosaic and the Web*. Chicago. Online verfügbar: `http://www.uic.edu/~cmsmcq/htmlmax.html`.

Ule, T. und Hinrichs, E. (2003): „Linguistische Annotation“. In: *Lobin und Lemnitzer* (2003b), S. 217–243. Erscheint.

Ule, T. und Müller, F. H. (2003): „KaRoPars: Ein System zur linguistischen Annotation großer Text-Korpora des Deutschen“. In: *Mehler und Lobin* (2003). Erscheint.

Volk, M. (1998): „Markup of a Test Suite with SGML“. In: Nerbonne, J., Hrsg. , *Linguistic Databases*, Nr. 77, CSLI Lecture Notes, S. 59–76. Cambridge: Cambridge University Press. Online verfügbar: `http://www.ifi.unizh.ch/CL/volk/papers/SGMLMarkup.ps.gz`.

Walker, D. (1999): „Taking Snapshots of the Web with a TEI Camera“. In: *Computers and the Humanities*, (33), 185–192.

Witt, A. (1999): „SGML und Linguistik“. In: *Lobin* (1999b), S. 121–154.

Witt, A. (2003): „Linguistische Informationsmodellierung mit XML“. In: *Mehler und Lobin* (2003). Erscheint.

Wolff, C. (2003): „Systemarchitekturen – Aufbau texttechnologischer Anwendungen“. In: *Lobin und Lemnitzer* (2003b), S. 165–192. Erscheint.