1 Texttechnologische Grundlagen

Georg Rehm

Die Texttechnologie ist ein junges Forschungsfeld, in dessen Zentrum die linguistisch motivierte Informationsanreicherung und Verarbeitung von Texten und Korpora mit standardisierten Auszeichnungssprachen steht. Eine zielgerichtete Definition ist aufgrund der zahlreichen, in konkreten Anwendungen potentiell beteiligten Disziplinen – u. a. Computer- und Korpuslinguistik, Text- und Hypertext-Theorie, Parsing, Text-Mining – nicht ohne weiteres möglich, d. h. die Texttechnologie umfasst nicht eine eindeutig bestimmbare Menge aufeinander aufbauender Theorien oder Methoden, sie ist vielmehr „wissenschaftlich begründete Praxis“ (Lobin and Lemnitzer, 2004a, S.1). Als kleinster gemeinsamer Nenner wird in allen texttechnologischen Anwendungen die Metasprache **XML** (*Extensible Markup Language*, Bray et al., 2008) eingesetzt, die die Definition beliebiger Auszeichnungssprachen erlaubt. XML ist eine formale Sprache zur Spezifizierung konkreter **Markup-Sprachen**, die wiederum zur **Auszeichnung** (auch: **Annotation**) arbiträrer Informationseinheiten in Texten und Datensammlungen eingesetzt werden können. In computerlinguistischen Anwendungen geht es hierbei u. a. um die Auszeichnung linguistischer Merkmale, die Verwendung von XML als Datenaustauschformat und den Einsatz *einer* Datengrundlage in *mehreren 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 konsequente **Trennung von Form und Struktur** durch die Einführung einer Abstraktionsebene, in der ein Textfragment von einem deklarativen Etikett wie z. B. `headline`, `absatz` oder `example` umschlossen wird, ohne dabei jedoch Textsatz- oder Layout-Anweisungen zu kodieren. Im Vordergrund steht also nicht die typografische Gestaltung einer Wortfolge, sondern die eindeutige Markierung ihrer logischen Funktion bzw. ihrer Semantik im Kontext eines spezifischen Dokumenttyps. Die **Auszeichnungselemente** – häufig schlicht Elemente genannt – einer Markup-Sprache bestehen aus einem Start- und einem End-Tag. SGML erlaubte das Wegfallen eines der beiden Tags, in XML kann eine verkürzte Schreibweise für Elemente ohne Inhalt benutzt werden (leere Elemente). Eine **Dokumentgrammatik** legt explizit die hierarchischen Kombinationsmöglichkeiten hinsichtlich der Strukturierung von Elementen fest, weshalb die in einem annotierten Dokument enthaltenen Elemente im graphentheoretischen Sinn einen Baum aufspannen (vgl. Abb. 1 auf S.3). Die Formalismen SGML und XML sind also selbst keine Auszeichnungssprachen, sondern **Metasprachen**, mit denen konkrete Auszeichnungssprachen definiert werden können. Die Weiterverarbeitung eines XML-annotierten Textes (auch: **Dokumentinstanz**) zu einem formatierten und publizierbaren Dokument erfolgt mit CSS oder XSL/XSLT (*Extensible Stylesheet Language*, *XSL Trans-*

formations, Clark, 1999; Kay, 2007). Die Auslagerung der Abbildung einzelner Tags auf korrespondierende Layout-Anweisungen ist der Grundstein des **Cross Media** bzw. **Single Source Publishing**. Hierunter versteht man die Möglichkeit, aus ein- und derselben annotierten Textquelle mit Hilfe unterschiedlicher **Stylesheets** z. B. eine für den Einsatz im WWW aufbereitete XHTML-Version und eine für den Druck optimierte PDF-Version generieren zu können.

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 einige der mittlerweile zahlreichen Standards dargestellt werden, die XML flankieren und zusätzliche Funktionalität bereit stellen. Abschnitt 1.4 geht auf den Einsatz von XML in computerlinguistischen Anwendungen ein, woraufhin Abschnitt 1.5 Lösungen zur Problematik der parallelen Annotation mehrerer Ebenen vorstellt.

1.1 HTML – Hypertext Markup Language

Webdokumente werden mit Hilfe der **Hypertext Markup Language** (Raggett et al., 1999; Pemberton, 2002) strukturiert und gestaltet – HTML stellt zugleich die bekannteste Auszeichnungssprache dar (siehe Unterkapitel **??**, *Das World Wide Web*). Version 4.01 des Standards 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>`, und *table data cells*, `<td>`) modelliert. Weitere Elemente erlauben z. B. die Auszeichnung von Überschriften, Listen und die Integration von Hyperlinks. In HTML werden unterschiedliche Auszeichnungsebenen vermischt, denn es existieren Elemente für strukturelles (z. B. `<h1>`, eine Überschrift erster Stufe), logisches (`<em>`, `<strong>` zur Markierung wichtiger Textteile), präsentationsorientiertes (`<i>`, `<b>` für Kursiv- bzw. Fettdruck), referentielles (`<a>`) und funktionales Markup (`<object>`, zur Einbettung externer Programme).

Die Dokumenttyp-Definitionen (DTD), d. h. die regelbasierten, formalen Definitionen, die die Namen und das Zusammenspiel von Elementen und Attributen spezifizieren, wurden bis HTML 4.01 als Anwendungen der Metasprache SGML definiert. Im Jahr 2000 wurde erstmals eine Neuformulierung auf der Grundlage von XML vorgenommen, um Kompatibilität mit dem XML-Paradigma zu gewährleisten. **XHTML** 1.0 (Pemberton, 2002) ist dabei nur der erste Schritt: Mit (X)HTML 5 und XHTML 2.0 sind bereits die nachfolgenden Standards in der Entwicklung, die unter anderem auf Modularität, Internationalisierung, deutlich flexiblere Möglichkeiten der Strukturierung von Texten und Repräsentation von Metadaten sowie bessere Adaptivität von Dokumenten an unterschiedliche Endgeräte abzielen.

1.2 XML – Extensible Markup Language

Bereits 1994 wurde angeregt, das fest vorgeschriebene, auf SGML beruhende statische Inventar von HTML-Elementen durch einen flexibleren Mechanismus

zu ergänzen, der die Definition beliebiger Web-bezogener Auszeichnungssprachen erlaubt. Gerade die mangelnde Erweiterbarkeit von HTML war für das **World Wide Web Consortium** (**W3C**, `http://www.w3.org`), das Standards im Umfeld des WWW konzipiert und verabschiedet, der Anlass, 1998 erstmalig die **Extensible Markup Language** (XML, Bray et al., 2008) zu spezifizieren, die eben diese Explizierung arbiträrer Informationen ermöglicht, indem die strikte Trennung von Dokumentform und Dokumentstruktur verfolgt wird. Der sehr komplexe Standard SGMLwurde mittlerweile fast vollständig von XML verdrängt, das eine Teilmenge des 1998 eingeführten WebSGML darstellt. Beibehalten wurde das Kernmerkmal von SGML: die Möglichkeit der Definition beliebiger Markup-Sprachen zur hierarchischen Strukturierung arbiträrer Informationseinheiten. Der Strukturaspekt bezieht sich dabei in vielen XML-basierten Markup-Sprachen – und 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 Rahmen einer texttechnologischen Anwendung z. B. die rhetorische Textstruktur auf der Grundlage der Rhetorical Structure Theory (RST, vgl. etwa Lobin, 1999a und Rehm, 1999), die Phrasenstruktur einzelner Sätze oder semantische Beziehungen wie etwa Koreferenz.

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

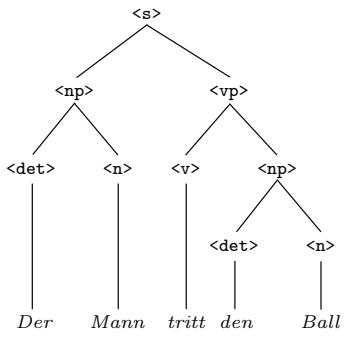

Abbildung 1: Eine XML-Dokumentinstanz am Beispiel der Annotation der Phrasenstruktur des Satzes „Der Mann tritt den Ball“

Abb. 1 zeigt mit einer **XML-Dokumentinstanz** ein Beispiel (nach Witt, 1999) sowie die von den XML-Elementen aufgespannte Baumstruktur, die hier der syntaktischen Struktur entspricht. Die zugehörige **Dokumenttyp-Definition** (**DTD**) folgt einer im XML-Standard festgelegten Syntax und enthält im Wesentlichen die Namen von Elementen und Attributen sowie die Kombinationsmöglichkeiten von Elementen, die durch **Inhaltsmodelle** spezifiziert werden.

```
<!ELEMENT s   (np, vp)>
<!ELEMENT np  (det, n)>
```

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

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

Die ersten sechs Deklarationen der DTD definieren, eingeleitet durch das Schlüsselwort `ELEMENT`, sechs **Auszeichnungselemente** und ihre Inhaltsmodelle; `s` kann in diesem Beispiel als Wurzelelement aufgefasst werden. Es fungiert als äußerstes Element einer Dokumentinstanz, das laut Inhaltsmodell in einer Instanz zunächst das Element `np` gefolgt von dem Element `vp` enthalten muss. 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 möglichen Vorkommen einzelner Elemente wird – ähnlich wie in regulären Ausdrücken – durch **Okkurrenzindikatoren** festgelegt. Im Inhaltsmodell von `s` befinden sich keine Okkurrenzindikatoren, weshalb in einer Instanz jeweils genau ein `np`- und ein `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. Durch Klammerung einzelner Teile eines Inhaltsmodells und 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 Inhaltsmodelle der **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 zusätzliche Informationen in einem Element hinterlegt werden – hierbei gilt die Faustregel, dass sich Attribute nur auf **Metadaten**, d. h. Informationen *über* die annotierten Daten, beziehen sollten (Maler and Andaloussi, 1996); Lehmberg and Wörner (2008) diskutieren die wesentlichen Aspekte und Standards bei der Auszeichnung linguistischer Korpora und Metadaten, Rehm et al. (2008) stellen eine konkrete Implementierung vor. In Attributlistendeklarationen (`ATTLIST`) wird zunächst festgelegt, auf welches Element sich ein Attribut bezieht. Im Beispiel werden die Attribute `kasus` und `agr` definiert, die für `np` bzw. `v` gelten. Es existieren unterschiedliche Attributtypen, die in einer Instanz u. a. die Eingabe eines beliebigen Textes als Wert erlauben. Das Beispiel zeigt jedoch den Aufzählungstyp, bei dem in der DTD die erlaubten Werte spezifiziert werden. Dabei gibt `#REQUIRED` an, dass ein Attribut obligatorisch ist (`#IMPLIED` markiert Optionalität). Neben den im Beispiel aufgeführten Syntaxelementen existieren noch Entitäten, die als Platzhalter eingesetzt werden können. Weiterhin sind bei der DTD-Erstellung verschiedene Ausnahmen zu beachten, z. B. sind mehrdeutige Inhaltsmodelle wie `((a, c) | (a, d))` verboten, können jedoch in den meisten Fällen umformuliert werden: `(a, (c | d))`.

1.2.1 Schemasprachen: Alternative Dokumentgrammatiken

Das syntaktische Inventar zur Spezifizierung von DTDs ist relativ eingeschränkt, weshalb das W3C mit **XML Schema** (Fallside et al., 2004) eine Spezifikation standardisiert hat, die präzisere und flexiblere Restriktionen in Instanzen ermöglicht. Neben einer groSSen Menge vordefinierter Datentypen gestattet XML Schema auch 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 zum Beispiel **Relax NG** (ISO/IEC 19757-2, 2003, `http://relaxng.org`) oder **Schematron** (ISO/IEC 19757-3, 2006, `http://www.schematron.com`) besteht darin, dass derartige Schemabeschreibungen ebenfalls in einer XML-Notation kodiert werden. Die proprietäre Syntax, die innerhalb von DTDs benutzt wird, wird also aufgegeben, damit Schemabeschreibungen – ebenso wie XML-Instanzen – mit beliebigen XML-Werkzeugen verarbeitet werden können.

1.2.2 Formale Eigenschaften von XML

Die Syntax einer formalen Sprache wird durch ein Startsymbol, Produktionsregeln, ein terminales sowie ein nichtterminales Vokabular definiert (vgl. Unterkapitel **??**). Analog zu dieser Terminologie können das Wurzelelement einer Dokumentgrammatik als Startsymbol und die einzelnen Elementdeklarationen als Syntaxregeln konzeptualisiert werden; Containerelementestellen das nichtterminale Vokabular dar, Datenelemente umfassen die terminalen Symbole. Diese Analogie zu kontextfreien Grammatiken lässt sich jedoch nur in terminologischen Aspekten finden, denn DTDs und die Schemasprachen erzeugen keine linearen Ketten, sondern hierarchisch angeordnete Baumstrukturen, weshalb sie **reguläre Baumgrammatiken** genannt werden (Lobin, 2004; Mönnich and Morawietz, 2004). Weil DTDs nicht die Benutzung abstrakter Symbole in Regeln erlauben, werden sie als **lokale Baumgrammatiken** bezeichnet.

1.3 Verarbeitung XML-annotierter Daten

Ein zentraler Aspekt im Umgang mit XML-Instanzen betrifft deren Verarbeitung mit **XML-Parsern** (z. B. expat, xmllint und Xerces), die nach dem gleichen Prinzip arbeiten wie Parser für natürliche Sprachen: Die Dokumentgrammatik fungiert als formale Grammatik, gegen die eine Dokumentinstanz als Eingabekette überprüft werden kann. Läuft dies fehlerfrei ab, 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 muss jedoch nicht notwendigerweise eine Dokumentgrammatik zu Grunde liegen, sie können von einem Parser auch isoliert verarbeitet werden, wobei dann jedoch nur überprüft werden kann, ob das Dokument der im XML-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 verschiedenen frei erhältlichen Werkzeugen möglich, so bieten etwa Firefox, Safari und Internet Explorer Möglichkeiten der Visualisierung der Baumstruktur, der integrierten XSLT-Transformation einer Instanz nach XHTML oder der Zuordnung eines **Cascading Style Sheets** (CSS, siehe Bos et al., 1998, 2007), um Elemente kontextabhängig formatieren zu können.

XSLT-Transformationenbasieren auf dem Prinzip der regelgesteuerten Überführung eines XML-Dokuments zur Umstrukturierung oder, dies stellt 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; Kay, 2007) und **XPath** (Clark and DeRose, 1999; Berglund et al., 2007) die *Extensible Stylesheet Language* (XSL) bildet. XPath übernimmt hierbei die Rolle eines Hilfsstandards, der die Navigation in der Baumrepräsentation eines Dokuments und die Adressierung sowie die Auswahl abstrakter Knoten (Elemente, Attribute, Text etc.) ermöglicht. Mit Hilfe von XPath-Ausdrücken ist es in einem **XSLT-Stylesheet** möglich, für einzelne Elemente einer Dokumentinstanz korrespondierende **Templates** (Transformationsregeln) zu spezifizieren. XSLT ist wiederum selbst als eine Markup-Sprache realisiert, d. h. XSLT-Stylesheets sind zugleich XML-Instanzen und können – wie alternative Schemabeschreibungen – zum Gegenstand XML-basierter Verarbeitungsprozesse werden. Zur Anwendung eines XSLT-Stylesheets wird ein **XSLT-Prozessor** wie z. B. Saxon, Xalan, Sablotron oder xsltproc benötigt.

1.3.2 Datenstrom- vs. baumbasierte Verarbeitung

Nahezu alle aktuellen Hochsprachen wie z. B. Java oder Perl bieten eine oder mehrere Schnittstellen zur Verarbeitung von XML-Dokumenten an. Üblicherweise wird hierbei entweder die SAX- oder die DOM-basierte Verarbeitung eingesetzt: **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 Ereignisse auslösen (*event-based processing*), die durch Funktionen abgefangen werden können. Da die Instanz nicht vollständig in den Speicher eingelesen werden muss, eignet sich SAX insbesondere zur Verarbeitung großer Datenmengen sowie für einfache Filterapplikationen. Im Gegensatz dazu erzeugt ein **DOM**-Prozessor (Document Object Model, Apparao et al., 2000; Hors et al., 2000, 2004) eine interne Baumrepräsentation (*tree-based processing*). Die unterschiedlichen Ebenen des DOM-Standards definieren 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 aus einer Datenbank). Neben SAX und DOM existiert ein dritter Ansatz, das Streaming API for XML. StAX ist jedoch nur für Java verfügbar.

1.3.3 Datenhaltung von XML-Instanzen

Zur Datenhaltung existieren verschiedene Möglichkeiten, wobei Dokumentinstanzen in der Regel innerhalb des Dateisystems gepflegt werden. 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 und technologisch reizvollste Möglichkeit betrifft den Einsatz nativer **XML-Datenbanken** wie zum Beispiel eXist, MonetDB, Qizx oder Tamino. Dieser neuartige Datenbanktyp, der eben nicht auf dem relationalen oder objektorientierten Paradigma basiert, speichert Instanzen in internen Datenstrukturen und unterstützt vielfältige Zugriffs- und Retrieval-Funktionen, die auf XPath, XQuery und jeweils eigenständigen Implementierungen der XQuery Update Facility basieren. Abseits von nativen XML-Datenbanken bieten alle großen relationalen Datenbanken (z. B. Oracle und IBM DB2) einen nativen XML-Datentyp an, der keine Umformung von XML-Instanzen in relationale Tabellenstrukturen voraussetzt.

1.3.4 Flankierende XML-Standards

Um XML gruppieren sich zahlreiche, teils verabschiedete, teils 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 (Bray et al., 2006). Hierzu werden die Namensräume sowie korrespondierende Präfixe deklariert, die in Elementen zur Referenzierung der Schemata benutzt werden (z. B. `<myns:doc xmlns:myns="http://example.org/myns">`). Zur Verknüpfung von Instanzen, aber auch zur Verbindung von Ressourcen im WWW wurden XML Pointer (XPointer), XML Base und die XML Linking Language (XLink) 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 standardisierte XML-Anwendung zur Beschreibung von Vektorgraphiken zur Verfügung. SVG bietet interessante Visualisierungsmöglichkeiten, 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 von Wolff (2004) aus computerlinguistischer Perspektive dargestellt. Aus computerlinguistischer Sicht von besonderem Interesse ist **XProc**: Mit Hilfe der XML

Pipeline Language können Operationen und Verarbeitungsschritte auf Gruppen von XML-Dokumenten definiert werden (Walsh et al., 2008).

In sprachtechnologischen Anwendungen werden häufig **RDF** (**Resource Description Framework**, `http://www.w3.org/RDF/`), **OWL** (**Web Ontology Language**, McGuinness and van Harmelen, 2004) und **XML Topic Maps** (XTM, Pepper and Moore, 2001) zur Modellierung von semantischen Netzen und Ontologien eingesetzt (siehe Unterkapitel **??**). RDF, konzipiert zur Repräsentation arbiträrer Metadaten, arbeitet mit den Objekttypen *Resources* (die Entität, über die eine Aussage getroffen wird), *Properties* (eine spezifische Eigenschaft der 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 Spezifizierung 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) belegt wird. Abb. 2 zeigt dieses Beispiel in erweiterter Form. RDF spezifiziert lediglich eine Syntax zur Beschreibung benannter Eigenschaften von Ressourcen sowie assoziierter Werte; zur Einbettung von RDF-Aussagen in XHTML-Dokumente wird derzeit an RDFa (RDF in attributes) gearbeitet. Die eigentliche Mächtigkeit von RDF wird jedoch erst durch **RDF Schema** (**RDFS**, Brickley and Guha, 2004) und OWL (McGuinness and van Harmelen, 2004) erreicht. RDFS und insbesondere OWL gestatten die Spezifizierung von RDF-Vokabularien durch die Definition typisierter Relationen und Eigenschaften mit Hilfe von Editoren wie z. B. Protégé, wobei auch Vererbungen von Eigenschaften modellierbar sind, so dass letztlich semantische Netze und komplexe Ontologien beschrieben werden können, auf die auch Inferenzverfahren anwendbar sind. Relevant ist in diesem Zusammenhang auch der ursprünglich für SGML entwickelte Standard **Topic Maps** (ISO/IEC 13250), für den mit XML Topic Maps (**XTM**, Pepper and Moore, 2001) auch ein XML-basiertes Serialisierungsformat vorliegt. In Topic Maps werden Knoten (Topics) mit Kanten (Associations) verbunden und können auch mit einem Typensystem versehen werden. Dieses abstrakte Wissen kann anschließend mit konkreten Instanzen (Occurrences) verbunden werden.

1.4 Texttechnologie und Computerlinguistik

In sprachtechnologischen Anwendungen nimmt XML mittlerweile einen zentralen Stellenwert ein – die **Annotation von Korpora** ist das prominenteste Beispiel (vgl. Unterkapitel **??**). Für diesen Zweck werden häufig die von der **Text Encoding Initiative** (**TEI**, `http://www.tei-c.org`, Burnard and Bauman, 2007) entwickelten Auszeichnungssprachen benutzt, die die Annotation unter-

```
<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

schiedlichster Textsorten (Gedichte, Dramen, historische Materialien, Lexika) auf verschiedenen Ebenen erlauben (Stührenberg, 2007): Zunächst können Metadaten von Dokumenten erfasst werden. Die zweite Ebene umfasst textuelle 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 (Witt, 2004). Ein wichtiger Aspekt betrifft die Zeichensatzkodierung: XML basiert auf **Unicode**, wodurch beliebige Alphabete repräsentierbar sind (Sasaki and Witt, 2004). Generell konzentriert sich die Anwendung von XML in sprachverarbeitenden Systemen auf den Bereich der textuellen Datenbanken (Lobin, 1999a) und die Anreicherung, Auswertung und Aufbereitung der Daten auf einer oder mehreren linguistischen Ebenen. Dabei geht es um die **manuelle oder automatische Annotation von Texten** (siehe z. B. Ule and Hinrichs, 2004; Ule and Müller, 2004) und die **Manipulation** sowie die **Verwendung des annotierten Materials** in konkreten Anwendungsszenarien (Lobin and Lemnitzer, 2004b; Mehler and Lobin, 2004).

Die automatische Annotation von Daten kann in den unterschiedlichsten Anwendungsszenarien und mit beliebigen computerlinguistischen Verfahren erfolgen (vgl. Kapitel **??**). Ein denkbares Szenario betrifft das Beispiel der Annotation von Sätzen mit Phrasenstruktur-Elementen (vgl. Abb. 1). Ein syntaktischer Parser für ein Fragment des Deutschen könnte derartige Annotationen aus der Analyse des Eingabesatzes erzeugen, die daraufhin automatisch mit annotierten Testsätzen verglichen werden können (vgl. Volk, 1998). In diesem speziellen Beispiel kodiert die Dokumentgrammatik Syntaxregeln (z. B. VP → V NP *), was z. B. bei der Implementierung und Evaluation von Chunk-Parsern ausgenutzt werden kann (vgl. Ule and Müller, 2004, siehe auch Abschnitt **??**). Zu diesem Zweck muss zunächst ein Korpus aufgebaut werden, wobei auch die typischen Vorverarbeitungsstufen Tokenisierung, Satzgrenzenidentifizierung und Part-of-Speech-Tagging einzusetzen sind. Derartige Anwendungsszenarien werden oftmals als Pipelines in generischen Architekturen wie zum Beispiel **GATE** (General Architecture for Text Engineering, `http://gate.ac.uk`) oder **UIMA** (Unstructured Information Management Architecture, `http://incubator.apache.org/uima/`) realisiert. Während die Formate

derartiger Architekturen meist auf effizient manipulierbaren Datenstrukturen basieren, ist es in der Regel doch immer möglich, an beliebigen Stellen etwa einer UIMA-Verarbeitungskette eine XML-Serialisierung der internen Datenstrukturen zu erzeugen, um sie z. B. auf fehlerhafte Annotationen zu überprüfen oder weiterführenden XML-Verarbeitungsschritten zur Verfügung zu stellen. Interessant bei der maschinellen Annotation linguistischer Informationen ist unter anderem die Möglichkeit, automatisch erzeugte Instanzen gegen eine DTD oder Schemabeschreibung zu validieren. Beim Parsing derartiger XML-Dateien gegen die von der Dokumentgrammatik erlaubten Strukturen resultieren fehlerhafte Annotationsalgorithmen unmittelbar in ungültigem Markup, was wiederum Fehlermeldungen des XML-Parsers bewirkt (Rehm, 1999), die z. B. zur Evaluation des Systems benutzt oder automatisch ausgewertet werden können, um dynamisch neue Annotationsregeln zu generieren.

1.5 Moderne Ansätze: Annotation auf mehreren Ebenen

Für nahezu alle Verwendungsfälle, in denen Sprach- oder Textdaten mit Informationen – üblicherweise linguistisch motivierten Analysen – angereichert werden, wird in der Computerlinguistik mittlerweile XML eingesetzt; Ide (2007) spricht sogar bereits von der „**Annotation Science**“, die als genuin linguistische Problematik markiert wird. Aufgrund des sprunghaft gestiegenen Interesses an texttechnologischen Methoden wurden vor einigen Jahren die konzeptionellen Grenzen XML-basierter Annotationen erkannt und in der Folgezeit mit unterschiedlichen Vorschlägen adressiert. Der traditionelle Ansatz, Elemente und Attribute in ein einzelnes XML-Dokument zu integrieren, wird als „**embedded XML**“ bezeichnet und ist in der Regel für die meisten computerlinguistischen Anwendungen ausreichend. Schwierigkeiten tauchen beispielsweise dann auf, wenn diskontinuierliche Konstituenten annotiert werden sollen: Derartige kreuzende Kanten sind in modernen Syntaxanalysen allgegenwärtig, können jedoch durch die von den XML-Elementen aufgespannte Baumstruktur nicht repräsentiert werden. Man kann sich hier zwar mit XML-Attributen und ihrem ID/IDREF-Mechanismus behelfen, doch ist diese Lösung weder flexibel noch für multiple Annotationen auf mehreren Ebenen geeignet. Aus diesen Gründen wird in aktuellen computerlinguistischen Annotationsprojekten oftmals der „**standoff XML**“-Ansatz verfolgt (Thompson and McKelvie, 1997): Ein zu annotierendes Dokument liegt in XHTML oder als Textdatei vor und wird *nicht* editiert. Stattdessen findet eine Basistokenisierung oder -segmentierung in einem XML-Dokument statt, das Annotationen über Pointer (z. B. zeichenbasierte Offsets) auf das Quelldokument bezieht. Sobald eine solche Tokenisierungsebene vorliegt, können sich weitere XML-Dokumente (mit Annotationen von Wortarten, Chunks, Koreferenz, semantischen Rollen etc.) auf diese Basisebene beziehen. Überlappende Kanten stellen in diesem Ansatz keine Schwierigkeit dar, jedoch machen es moderne linguistische Analysen in der Regel notwendig, das XML-Paradigma mit der Idee des einzelnen Dokuments zu verlassen (doch siehe Stührenberg and Goecke, 2008), wodurch unter anderem beim Erzeugen, Editieren, Validieren und Querying solcher standoff-XML-Dokumente ein im-

menser Mehraufwand entsteht (Rehm et al., 2008), für den noch keine standardisierte Lösung existiert. Zu den aktuellen Vorschlägen zählen NITE (Carletta et al., 2003), Annolab (Eckart and Teich, 2007), die Datenbank ANNIS mit dem Format PAULA (Dipper et al., 2007), die Datenbank SPLICR mit dem Format GENAU (Wörner et al., 2006; Witt et al., 2007; Rehm et al., 2008) sowie insbesondere das Linguistic Annotation Framework (LAF, Ide et al., 2003) mit verschiedenen assoziierten Initiativen wie z. B. dem Data Category Registry (`http://www.isocat.org`).

Neben der Konzipierung und Implementierung konkreter texttechnologischer Lösungen stehen einige weitere Aspekte derzeit im Mittelpunkt der Diskussion. Hierzu zählen die Problematik der Nachhaltigkeit und Interoperabilität linguistischer Ressourcen (für einfachen Datenaustausch) sowie Fragen der nationalen und internationalen Standardisierung von Auszeichnungssprachen für die unterschiedlichsten computerlinguistischen Zwecke (Zinsmeister et al., 2008).

1.6 Literaturhinweise

Die Spezifikationen von XML sowie den beteiligten Standards pflegt das World Wide Web Consortium (W3C, `http://www.w3.org`). Wichtige Ressourcen sind `http://xml.coverpages.org` sowie die Proceedings der Konferenzreihe „Extreme Markup Languages“ (`http://www.extrememarkup.com`), die seit 2008 „Balisage: The Markup Conference“ (`http://www.balisage.net`) heißt. Die computerlinguistische und texttechnologische Literatur ist mittlerweile kaum noch überschaubar. Nach wie vor relevant sind die in Lobin (1999b) und Mehler and Lobin (2004) publizierten Beiträge, die Verknüpfungen von texttechnologischen und computerlinguistischen Methoden darstellen. Lobin and Lemnitzer (2004b) führen erstmals in die unterschiedlichen Facetten der Texttechnologie ein. Aktuelle Forschungsansätze im Bereich der Texttechnologie werden in den Zeitschriften *Literary and Linguistic Computing* sowie *Language Resources and Evaluation*, im Rahmen der jährlich stattfindenden Konferenz *Digital Humanities*, bei den *Linguistic Annotation Workshops* (LAW) sowie den verschiedenen Veranstaltungen aus dem TEI-Umfeld vorgestellt.

Literatur

Apparao, V., S. Byrne, M. Champion, S. Isaacs, I. Jacobs, A. L. Hors, G. Nicol, J. Robie, R. Sutor, C. Wilson, and L. Wood (2000, September). Document Object Model (DOM) Level 1 Specification (Second Edition). Technische Spezifikation, W3C.

Berglund, A., S. Boag, D. Chamberlin, M. F. Fernandez, M. Kay, J. Robie, and J. Siméon (2007, Januar). XML Path Language (XPath) 2.0. Technische Spezifikation, W3C. `http://www.w3.org/TR/xpath20/`.

Bos, B., T. Çelik, I. Hickson, and H. W. Lie (2007, Juli). Cascading Style Sheets, Level 2 Revision 1 – (CSS 2.1) Specification. Candidate Recommendation, W3C. `http://www.w3.org/TR/CSS`.

Bos, B., H. W. Lie, C. Lilley, and I. Jacobs (1998, Mai). Cascading Style Sheets, Level 2 – CSS2 Specification. Technische Spezifikation, W3C. http://www.w3.org/TR/REC-CSS2/.

Bray, T., D. Hollander, A. Layman, and R. Tobin (2006, August). Namespaces in XML 1.0 (Second Edition). Technische Spezifikation, W3C. http://www.w3.org/TR/xml-names/.

Bray, T., J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau (2008, November). Extensible Markup Language (XML) 1.0 (Fifth Edition). Technische Spezifikation, W3C. http://www.w3.org/TR/2008/REC-xml-20081126/.

Brickley, D. and R. Guha (2004, Februar). RDF Vocabulary Description Language 1.0: RDF Schema. Technische Spezifikation, W3C. http://www.w3.org/TR/rdf-schema/.

Burnard, L. and S. Bauman (Eds.) (2007). *TEI P5: Guidelines for Electronic Text Encoding and Interchange*. Text Encoding Initiative Consortium.

Carletta, J., J. Kilgour, T. J. O'Donnell, S. Evert, and H. Voormann (2003). The NITE Object Model Library for Handling Structured Linguistic Annotation on Multimodal Data Sets. In *Proceedings of the EACL Workshop on Language Technology and the Semantic Web (3rd Workshop on NLP and XML)*.

Clark, J. (1999, November). XSL Transformations (Version 1.0). Technische Spezifikation, W3C. http://www.w3.org/TR/xslt/.

Clark, J. and S. DeRose (1999, November). XML Path Language (XPath) – Version 1.0. Technische Spezifikation, W3C. http://www.w3.org/TR/xslt/.

Dipper, S., M. Götze, U. Küssner, and M. Stede (2007). Representing and Querying Standoff XML. See Rehm et al. (2007), pp. 337–346.

Eckart, R. and E. Teich (2007). An XML-Based Data Model for Flexible Representation and Query of Linguistically Interpreted Corpora. See Rehm et al. (2007), pp. 327–336.

Fallside, D. C., P. Walmsley, H. S. Thompson, D. Beech, M. Maloney, N. Mendelsohn, P. V. Biron, and A. Malhotra (2004, Oktober). XML Schema Second Edition. Technische Spezifikation, W3C. Besteht aus Part 0 (Primer), Part 1 (Structures), Part 2 (Datatypes). http://www.w3.org/TR/xmlschema-2/.

Hors, A. L., P. L. Hégaret, L. Wood, G. Nicol, J. Robie, M. Champion, and S. Byrne (2000, November). Document Object Model (DOM) Level 2 Core Specification. Technische Spezifikation, W3C.

Hors, A. L., P. L. Hégaret, L. Wood, G. Nicol, J. Robie, M. Champion, and S. Byrne (2004, April). Document Object Model (DOM) Level 3 Core Specification. Technische Spezifikation, W3C.

Ide, N. (2007). Annotation Science: From Theory to Practice and Use. See Rehm et al. (2007), pp. 1–7.

Ide, N., L. Romary, and E. de la Clergerie (2003). International standard for a linguistic annotation framework. In *Proceedings of HLT-NAACL'03 Workshop on The Software Engineering and Architecture of Language Technology*.

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

ISO/IEC 19757-2 (2003). Information technology – Document Schema Definition Language (DSDL) – Part 2: Regular-grammar-based validation – RELAX NG (ISO/IEC 19757-2). Internationaler Standard, International Organization for Standardization, Geneva.

ISO/IEC 19757-3 (2006). Information technology – Document Schema Definition Language (DSDL) – Part 3: Rule-based validation – Schematron. Internationaler standard, International Organization for Standardization, Geneva. `http://standards.iso.org/ittf/PubliclyAvailableStandards/c040833_ISO_IEC_19757-3_2006(E).zip`.

Kay, M. (2007, Januar). XSL Transformations (Version 2.0). Technische Spezifikation, W3C. `http://www.w3.org/TR/xslt20/`.

Lehmberg, T. and K. Wörner (2008). Annotation Standards. See Lüdeling and Kytö (2008), pp. 484–501.

Lobin, H. (1999a). Intelligente Dokumente – Linguistische Repräsentation komplexer Inhalte für die hypermediale Wissensvermittlung. See Lobin (1999b), pp. 155–178.

Lobin, H. (Ed.) (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. (2004). Textauszeichnung und Dokumentgrammatiken. See Lobin and Lemnitzer (2004b), pp. 51–82.

Lobin, H. and L. Lemnitzer (2004a). Text(e) technologisch. See Lobin and Lemnitzer (2004b), pp. 1–9.

Lobin, H. and L. Lemnitzer (Eds.) (2004b). *Texttechnologie – Anwendungen und Perspektiven*. Stauffenburg Handbücher. Tübingen: Stauffenburg.

Lüdeling, A. and M. Kytö (Eds.) (2008). *Corpus Linguistics*. Handbücher zur Sprach- und Kommunikationswissenschaft (HSK). Berlin, New York: de Gruyter.

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

McGuinness, D. L. and F. van Harmelen (2004, Februar). OWL Web Ontology Language – Overview. Technische Spezifikation, W3C. `http://www.w3.org/TR/2004/REC-owl-features-20040210/`.

Mehler, A. and H. Lobin (Eds.) (2004). *Automatische Textanalyse – Systeme und Methoden zur Annotation und Analyse natürlichsprachlicher Texte*. Wiesbaden: Verlag für Sozialwissenschaften.

Mönnich, U. and F. Morawietz (2004). Formale Grundlagen. See Lobin and Lemnitzer (2004b), pp. 109–141.

Pemberton, S. (2002, August). XHTML 1.0: The Extensible Hypertext Markup Language (Second Edition). Technische Spezifikation, W3C. `http://www.w3.org/TR/xhtml1/`.

Pepper, S. and G. Moore (2001, Juni). XML Topic Maps (XTM) 1.0. Technische Spezifikation, TopicMaps.Org. `http://www.topicmaps.org/xtm/1.0/`.

Raggett, D., A. L. Hors, and I. Jacbos (1999, Dezember). HTML 4.01 Specification. Technische Spezifikation, W3C. `http://www.w3.org/TR/html401/`.

Rehm, G. (1999). Automatische Textannotation – Ein SGML- und DSSSL-basierter Ansatz zur angewandten Textlinguistik. See Lobin (1999b), pp. 179–195.

Rehm, G., R. Eckart, C. Chiarcos, and J. Dellert (2008, Mai). Ontology-Based XQuery'ing of XML-Encoded Language Resources on Multiple Annotation Layers. In *Proceedings of the 6th Language Resources and Evaluation Conference (LREC 2008)*, Marrakech, Morocco.

Rehm, G., O. Schonefeld, A. Witt, C. Chiarcos, and T. Lehmberg (2008, September). SPLICR: A Sustainability Platform for Linguistic Corpora and Resources. In A. Storrer, A. Geyken, A. Siebert, and K.-M. Würzner (Eds.), *KONVENS 2008 (Konferenz zur Verarbeitung natürlicher Sprache) – Textressourcen und lexikalisches Wissen*, Berlin, pp. 86–95.

Rehm, G., O. Schonefeld, A. Witt, T. Lehmberg, C. Chiarcos, H. Bechara, F. Eishold, K. Evang, M. Leshtanska, A. Savkov, and M. Stark (2008, Mai). The Metadata-Database of a Next Generation Sustainability Web-Platform for Language Resources. In *Proceedings of the 6th Language Resources and Evaluation Conference (LREC 2008)*, Marrakech, Morocco.

Rehm, G., A. Witt, and L. Lemnitzer (Eds.) (2007). *Datenstrukturen für linguistische Ressourcen und ihre Anwendungen – Data Structures for Linguistic Resources and Applications: Proceedings of the Biennial GLDV Conference 2007*. Tübingen: Gunter Narr.

Sasaki, F. and A. Witt (2004). Linguistische Korpora. See Lobin and Lemnitzer (2004b), pp. 195–216.

Stührenberg, M. (2007). Texttechnological Standards – An Overview. See Rehm et al. (2007), pp. 157–166.

Stührenberg, M. and D. Goecke (2008, August). SGF – An Integrated Model for Multiple Annotations and its Application in a Linguistic Domain. In *Proceedings of Balisage: The Markup Conference*.

Thompson, H. S. and D. McKelvie (1997). Hyperlink Semantics for Standoff Markup of Read-Only Documents. In *Proceedings of SGML Europe '97: The next decade – Pushing the Envelope*, Barcelona, pp. 227–229.

Ule, T. and E. Hinrichs (2004). Linguistische Annotation. See Lobin and Lemnitzer (2004b), pp. 217–243.

Ule, T. and F. H. Müller (2004). KaRoPars: Ein System zur linguistischen Annotation groSSer Text-Korpora des Deutschen. See Mehler and Lobin (2004).

Volk, M. (1998). Markup of a Test Suite with SGML. In J. Nerbonne (Ed.), *Linguistic Databases*, Number 77 in CSLI Lecture Notes, pp. 59–76. Cambridge: Cambridge University Press.

Walsh, N., A. Milowski, and H. S. Thompson (2008, November). XProc: An XML Pipeline Language. Technische Spezifikation (Candidate Recommendation), W3C. `http://www.w3.org/TR/xproc/`.

Witt, A. (1999). SGML und Linguistik. See Lobin (1999b), pp. 121–154.

Witt, A. (2004). Linguistische Informationsmodellierung mit XML. See Mehler and Lobin (2004).

Witt, A., O. Schonefeld, G. Rehm, J. Khoo, and K. Evang (2007, August). On the Lossless Transformation of Single-File, Multi-Layer Annotations into Multi-Rooted Trees. In B. T. Usdin (Ed.), *Proceedings of Extreme Markup Languages 2007*, Montréal, Canada.

Wolff, C. (2004). Systemarchitekturen – Aufbau texttechnologischer Anwendungen. See Lobin and Lemnitzer (2004b), pp. 165–192.

Wörner, K., A. Witt, G. Rehm, and S. Dipper (2006, August). Modelling Linguistic Data Structures. In B. T. Usdin (Ed.), *Proceedings of Extreme Markup Languages 2006*, Montréal, Canada.

Zinsmeister, H., A. Witt, S. Kübler, and E. Hinrichs (2008). Linguistically Annotated Corpora: Quality Assurance, Reusability and Sustainability. See Lüdeling and Kytö (2008), pp. 759–776.

Index

Annotation, *siehe* Auszeichnung
Annotation Science, 10
Auszeichnung, 1, 2, 4
    manuelle, 9
    maschinelle, 9
    von Korpora, 8, 9
Auszeichnungselement, 1, 3
Auszeichnungssprache, *siehe* Markup-Sprache

Baum, 1, 3, 5, 6, 10
Baumgrammatik
    lokale, 5
    reguläre, 5
Browser, 6

Cascading Style Sheet, *siehe* CSS
Cross Media Publishing, 2
CSS, 1, 6

DocBook, 6
Document Object Model, *siehe* DOM
Dokumentgrammatik, *siehe* DTD
Dokumenttyp, 1
Dokumenttyp-Definition, *siehe* DTD
DOM, 6
    DOM-Prozessor, 6
DTD, 2–6, 10

Extensible Markup Language, *siehe* XML
Extensible Stylesheet Language, *siehe* XSL

GATE, 9
General Architecture for Text Engineering, *siehe* GATE
Grammatik
    kontextfreie, 5
    lokale Baumgrammatik, 5
    reguläre Baumgrammatik, 5

HTML, 2, 6, 7, 10
    HTML-Dokument, 8
Hypertext Markup Language, *siehe* HTML

Informationsmodellierung, 1

Korpora
    Annotation, 4, 8, 9

Link, 7
    bidirektionaler, 7
    typisierter, 7

Markup-Sprache, 1
Metadaten, 2, 4, 8, 9
Metasprache, 1, 2

Namespaces, 7

Ontologie, 8
OWL, 8

Parser, 5, 9
    Chunk-Parser, 9
    XML-Parser, 5, 10
Phrasenstruktur, 9

RDF, 8
    RDF-Vokabular, 8
RDF Schema, *siehe* RDFS
RDFS, 8
Relax NG, 5
Resource Description Framework, *siehe* RDF
Rhetorical Structure Theory, *siehe* RST
RST, 3

SAX, 6
Scalable Vector Graphics, *siehe* SVG
Schemasprache, 5, 6
Schematron, 5
Semantisches Netz, 8
SGML, 1, 3
    WebSGML, 3
Simple API for XML, *siehe* SAX
Simple Object Access Protocol, *siehe* SOAP
Single Source Publishing, 2

SOAP, 7
Standard Generalized Markup Language, *siehe* SGML
Startsymbol, 5
StAX, 7
Streaming API for XML, *siehe* StAX
SVG, 7
Syntax
  Syntaxregel, 5, 9

TEI, 8
Template, 6
Textsorte, 9
Texttechnologie, 1, 8, 10
Textuelle Datenbanken, 9
Topic Maps, 8
Trennung von Form und Struktur, 1, 3

UIMA, 9
Unicode, 9
Unstructured Information Management Architecture, *siehe* UIMA

Validität, 5
Vererbung, 8
Vokabular
  nichtterminales, 5
  RDF-Vokabular, 8
  terminales, 5

W3C, 3, 5, 7, 11
  Standard, 3, 5, 7
Web Ontology Language, *siehe* OWL
Web Services, 7
WebSGML, 3
Wohlgeformtheit, 6
World Wide Web, *siehe* WWW
World Wide Web Consortium, *siehe* W3C
WWW, 2, 3, 7

XHTML, 2, 6–8, 10
  XHTML-Dokument, 8
XML, 1–10
  Betrachtung, 6
  Containerelement, 4, 5
  Datenelement, 4, 5
  DocBook, 6
  Dokumentinstanz, 1–7
  Embedded XML, 10
  Inhaltsmodell, 3–5
  Konnektor, 4
  Okkurrenzindikator, 4, 5
  Standoff XML, 10
  Validität, 5
  Web Services, 7
  Wohlgeformtheit, 6
  Wurzelelement, 4
  XLink, 7
  XML Schema, 5
  XML-Anwendung, 7
  XML-Datenbank, 7, 11
  XML-Parser, 5, 10
  XML-Werkzeug, 5
  XPath, 6
XML Pipeline Language, *siehe* XProc
XML Topic Maps, *siehe* XTM
XProc, 7
XSL, 1, 6
XSL Transformations, *siehe* XSLT
XSL-FO, 6
XSLT, 1, 6, 7
  Template, 6
  XSLT-Prozessor, 6
XTM, 8

Zeichensatzkodierung, 9