Validierung von XML – Dokumenten

Ausarbeitung Multimedia- und Webtechnologien

Prof. Gössner

WS 2005 / 2006

13.12.2005

von:

Daniela Nolte

Johannes Koop

Christoph Ungermann

Inhalt:

(DN) = ausgearbeitet von Daniela Nolte
(CU) = ausgearbeitet von Christoph Ungermann
(JK) = ausgearbeitet von Johannes Koop

1. Einführung / Definitionen

1.1 Validierung/ Plausibilisierung

Die Plausibilisierung oder Plausibilitätsprüfung bezeichnet die Überprüfung von Ergebnissen, die durch komplexe Prozesse ermittelt wurden hinsichtlich ihrer Einfügung in den angenommenen Ergebniserwartungsbereich. So können falsche Ergebnisse, zum Beispiel verursacht durch Rechenfehler, vermieden werden. In der Softwaretechnik bezeichnet Validierung oder Plausibilisierung (auch genannt „Santity Check“) die Kontrolle eines konkreten Wertes darauf, ob er zu einem bestimmten Datentyp gehört, in einem vorgegebenen Wertebereich oder in einer vorgegebenen Wertemenge liegt. Sie ist ein wichtiger Aspekt in der Qualitätssicherung, der sicherstellen soll, dass ein implementiertes Programm den vorher aufgestellten Anforderungen genügt. Die meisten Programmfehler und Sicherheitsprobleme sind letztlich auf fehlende Plausibilisierung von Eingabewerten zurückzuführen.
Für die Validierung gilt die Goldene Regel: „never trust the user“ (traue niemals dem Benutzer).

➔ Die Validierung ist ein pragmatischer Schutz gegen fehlerhafte Benutzung und eine Überprüfung auf die korrekte Eingabe von Daten durch den Benutzer!

1.2 XML

XML ist eine Metasprache*, mit der es möglich ist, Auszeichnungssprachen für Dokumente zu erzeugen. Eine bekannte Auszeichnungssprache** für das WWW ist HTML und HTML kann in XML formuliert werden. Im Unterschied zu HTML sind bei XML die Auszeichnungselemente (tags) jedoch nicht festgelegt. Tags werden durch spitze Klammern gekennzeichnet. Die Namen der Tags können im Unterschied zu

HTML vom Anwender selbst festgelegt, bzw. frei erfunden werden. Sie können so gewählt werden, dass sie die Bedeutung des Inhalts ausdrücken, z.B. <buchtitel>.

* Wissenschaftliches Sprachsystem; Sprache mit der eine andere Sprache erklärt oder definiert wird, mit der also die Regeln für diese Sprache festgelegt werden

** Auch Markup Language. Dient zur Beschreibung von Daten oder des Verfahrens, das zur Darstellung nötig ist. Bei einer Auszeichnungssprache werden die Eigenschaften, Zugehörigkeiten und Verfahren von bestimmten Wörtern, Sätzen und Abschnitten eines Textes beschrieben, meist in dem sie mit Tags markiert werden. Beispiel: die in SGML oder XML definierten Sprachen HTML, MathML, SGV, usw.

Die Extensible Markup Language (XML) ist ein Standard zur Erstellung von maschinen- und menschenlesbarer Dokumente in Form einer Baumstruktur.

XML definiert dabei die Regeln für den Aufbau solcher Dokumente und ist damit ein Standard zur Definition von beliebigen, in ihrer Grundstruktur jedoch stark verwandten Auszeichnungssprachen. Ein XML- Element kann ganz unterschiedliche Daten enthalten und beschreiben, als prominentes Beispiel „Text“, aber auch Grafiken oder abstraktes Wissen.

XML wurde von einer Arbeitsgruppe des World Wide Web Consortiums (W3C) entwickelt, um Daten bzw. Dokumente jeglicher Art strukturiert abzubilden. XML soll leicht im Internet zu nutzen sein und von vielen Anwendungen unterstützt werden. Das universelle Format von XML stellt sicher, dass Dokumente plattform- und anwendungsübergreifend verwendet werden können.

Anders als bei den anderen Auszeichnungssprachen werden bei XML die Daten, die Anweisungen und die Formatierung getrennt. Dies ermöglicht das Nutzen der Daten von verschiedenen Anwendungsprogrammen, zum Beispiel zum Datenaustausch oder als Schnittstelle für verschiedene Darstellungsformate, ohne das eigentliche Dokument zu verändern.

Um dies zu ermöglichen, werden XML- Dokumente nach präzisen Regeln aufgebaut. XML besitzt zwei Klassen der Korrektheit von Dokumenten. Die erste Klasse bilden die wohlgeformten Dokumente. Diese Dokumente halten die Syntax von XML ein. Aufbauend darauf existieren die gültigen Dokumente, welche zusätzlich den Regeln einer vorgegebenen Dokumenttyp- Deklaration entsprechen.

1. Wohlgeformtheit (well-formed)

Das Dokument hält sämtliche Regeln von XML ein. Wohlgeformtheit beschreibt eine Reihe von Bedingungen und Regeln, die als Mindestanforderungen erfüllt sein müssen um ein XML- Dokument mit XML- Werkzeugen verarbeiten zu können.

- Dokumente bestehen aus Tags und Inhalt

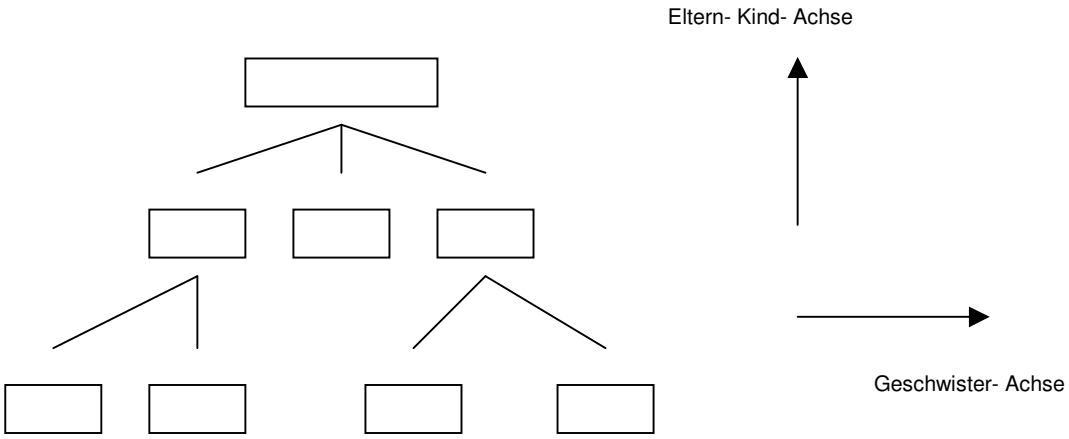
- Tags müssen geschlossen werden. Beispielsweise muss für jedes öffnende Element (z.B. <item>) ein schließendes Element (</item>) vorhanden sein.
- Tags müssen korrekt geschachtelt sein, es dürfen keine Überlappungen auftreten. Es muss darauf geachtet werden, dass die Auszeichnungen abgeschlossen werden, bevor eine übergeordnete Auszeichnung geschlossen wird.
- Attributwerte müssen in doppelten oder einfachen Anführungszeichen stehen
- Elementnamen dürfen nicht mit der Zeichenfolge xml und nicht mit Ziffern beginnen

Beispiel:

```
<Kapitel>
  <titel> Einführung in XML </titel>
    <Vorspann> XML ist eine Metasprache zur Definition von Auszeichnungs-
      sprachen</Vorspann>
    <Absatz> Wer mit XML arbeiten will muss sich mit folgenden Themen be-
      schäftigen</Absatz>
</Kapitel>
```

Zur Wohlgeformtheit gehört auch, dass jedes XML- Dokument ein Wurzelelement (root Tag) hat, welches das gesamte Dokument umschließt. Es definiert die Standardeinstellungen für alle anderen Elemente, z.B. die Einstellung der Auflösung auf Computerbildschirmen. Alle Auszeichnungen, die im inneren des Dokuments angewendet werden, sind hierarchisch in das Wurzelelement eingeschlossen, entsprechend dem obigen Beispiel.

Stellt man sich das Wurzelelement als Wurzel eines Stammbaumes vor, so sind alle Elemente im Dokument Nachfahren (Nachkommen) der Wurzel. Im XML – Jargon wird deshalb auch von Eltern- Elementen, Kind- Elementen, Geschwister- Elementen etc. gesprochen.



2. Gültigkeit (valid)

Damit ein XML- Parser* überprüfen kann, ob ein Dokument nicht nur wohlgeformt, sondern auch gültig ist, kann zu jedem XML- Dokument eine Dokument Type Deklaration (DTD) angegeben werden.
Soll XML für den Datenaustausch verwendet werden, ist es von Vorteil, wenn das Format mittels einer Grammatik (z.B. einer DTD) definiert ist. Ein XML- Dokument, welches wohlgeformt ist und ein durch eine Grammatik beschriebenes Format einhält, heißt gültig (valid).

XML wird heutzutage meistens dazu verwendet, Datenstrukturen anwendungsunabhängig zu definieren und auszutauschen. Eine zentrale Rolle spielt die Überprüfung, ob die Dokumente den Regeln einer DTD entsprechen. Die Aufgabe der Überprüfung liegt beim XML- Parser, der Dokumente einliest und einer verarbeitenden Anwendung zur Verfügung stellt.

Obwohl der Vorgänger von XML, SGML, bereits weitaus umfangreicher war, kam es nie zu einer breiten Akzeptanz in der Öffentlichkeit. Der Grund dafür liegt in der Komplexität von SGML, die die Softwareentwicklung stark erschwert. Die Komplexität von SGML und XML kann mit der Pareto- Verteilung beschrieben werden: obwohl XML nur ca. 20 % Komplexität von SGML hat, können ca. 80 % der Anwendungsfälle abgedeckt werden. Der Bedarf nach einem unbeschränkten weltweiten Informationsaustausch und die Popularität von HTML brachten das deutlich einfachere XML hervor.

* Programme und Programmteile, die XML- Daten auslesen, interpretieren und ggf. auf Gültigkeit prüfen.

1.3 DTD (Document Type Definition)

Soll das XML- Dokument nicht nur wohlgeformt sein, sondern soll zusätzlich festgelegt werden, welche Elemente, Attribute und welche Schachtelungen erlaubt sind, so muss ein eigener XML- Dokumenttyp beschrieben werden. Jeder dieser Dokumenttypen ist durch eine genau festgelegte Menge an Element- Typen und zugeordneten Attribut- Typen charakterisiert. Bei der Definition von jedem Element-Typ wird zusätzlich festgelegt, welche Typen von Unterelementen erlaubt sind, bzw. ob Text erlaubt ist. Eine solche Definition eines Dokument- Typs deklariert also, welche zusätzlichen Einschränkungen ein bereits wohlgeformtes XML- Dokument erfüllen muss, damit es zusätzlich als „Dokument dieses Dokumenttyps“ betrachtet werden darf. Mit einer Elementtyp- Deklaration werden ein Element und sein möglicher Inhalt definiert. In einem validen XML- Dokument dürfen nur Elemente vorkommen, die in der DTD definiert sind. Der Inhalt eines Elements kann durch die Angabe anderer Elementnamen und durch einige Schlüsselwörter und Zeichen angegeben werden, z.B. EMPTY für „keinen Inhalt“, ANY für „beliebigen Inhalt“, Runde Klammern zum Gruppieren,...).

Die Document type definition ist sozusagen eine Beschreibung eines XML-Dokuments, die zusammen mit XML standardisiert wurde.
In einer DTD werden Elemente (z.B. XML- Elemente), Attribute (z.B. XML- Attribute), Entitäten (z.B. XML- Entitäten) und Besonderheiten der Syntax- Verwendung (z.B. Abkürzungen oder lokale Umdefinitionen von Zeichen) definiert. Konkret heißt das, dass in einer DTD die Reihenfolge der Elemente oder die Art des Inhalts festgelegt wird.

Die Grammatikregeln der DTD können dabei sowohl innerhalb des XML-Dokumentes (interne DTD) als auch in einer externen Datei (externe DTD) angegeben werden.
Eine interne DTD befindet sich, wie der Name schon sagt, innerhalb des XML-Dokumentes, für das sie erstellt wurde. Eine interne DTD hat den gravierenden Nachteil, dass sie für jedes einzelne XML- Dokument neu geschrieben werden muss. Möchte man aber mehr als ein XML- Dokument nach den festgelegten Regeln

erstellen, dann bietet sich eher eine externe DTD an, nach der sich dann mehrere XML- Dokumente richten können. Dazu wird die externe DTD in einer separaten Textdatei gespeichert und mit dem XML- Dokument verknüpft, bzw. per Verweis eingebunden.
Neben den privaten DTD´s gibt es auch öffentliche DTD´s. Sie werden zum Beispiel von Verbänden, Industriezweigen oder Ähnlichem veröffentlicht, um XML-Dokumente einheitlich erstellen zu können.

Ein Beispiel: Ein Verleger könnte den Wunsch haben, dass Autoren bestimmte Formatvorschriften verwenden, weil es dadurch einfacher wird, das Layout von Büchern zu erstellen. Ein Autor mag es vorziehen, sein Werk als ein Fließtext zu verfassen, ohne sich über passende Schriftarten für den Kapitelanfang oder für Zwischenüberschriften innerhalb eines Kapitels Gedanken machen zu müssen. Falls der Autor in XML schreibt, ist es für den Verleger leicht zu prüfen, ob der Autor dem vorgeschriebenen Format gefolgt ist, das durch die DTD spezifiziert wird, und wo und auf welche Weise der Autor von dem Format abgewichen ist. Dies ist viel einfacher als einen Lektor damit zu beauftragen, Dokumente zu lesen und Abweichungen vom Format herauszufinden.

Mit DTD´s kann auch sichergestellt werden, dass verschiedene Leute und Programme ihre Dateien gegenseitig lesen können. Wenn sich beispielsweise Chemiker gemeinsam auf eine einzige DTD für die grundlegende chemische Notation einigen, möglicherweise durch Vermittlung eines Fachverbands, dann können sie sicher sein, dass sie ihre Fachaufsätze gegenseitig (maschinell) lesen und verstehen können. Die DTD legt genau fest, was in einem Dokument vorkommen darf und was nicht, und sie legt ebenfalls fest, dass Erweiterungen ungültig sind, die über die Definition in der DTD hinausgehen. Deshalb tragen DTD´s dazu bei, Softwareanbieter daran zu hindern, offene Protokolle für sich zu vereinnahmen und zu erweitern, um Benutzer an ihre firmenspezifische Software zu binden.

- ➔ Man kann mit Hilfe der DTD Regeln für ein XML- Dokument bestimmen. Diese Regeln bestimmen, welche Daten enthalten sein dürfen und welche nicht. Die DTD ist konservativ, d.h. alles was nicht ausdrücklich erlaubt ist, ist verboten.

1.4 XML- Schemata

DTD´s besitzen eine eigene nicht XML konforme Syntax und unterstützen Datentypen (z.B. string, decimal, boolean, date, time) nur in geringem Umfang. Das W3C hat deshalb den XML Schema Standard entwickelt, der seit Mai 2001 als stabiler Standard gilt. Wie eine DTD kann das Schema die Struktur eines XML-Dokuments beschreiben. Darüber hinaus wird eine große Anzahl von Datentypen unterstützt.

Vermutlich werden DTD´s irgendwann vollständig von XML- Schemata abgelöst. Allerdings sind XML- Schemata durch ihre erweiterten Möglichkeiten wesentlich komplexer und nicht so einfach ohne Hilfsmittel auszuwerten. Ein konkretes XML-Schema wird auch als eine XSD (XML- Schema- Definition) bezeichnet.

XML- Schema ist ein neuer Ansatz, ein XML- Dokument auf seine Gültigkeit zu prüfen. Ein XML- Dokument, welches gegen ein XML- Schema validiert, wird schema- valid (schema- gültig) genannt. Ein XML- Schema stellt wie eine DTD eine Beschreibung der Struktur eines XML- Dokuments dar, kann aber auch den Inhalt von Elementen und Attributen einschränken.
Ein XML- Schema ist auch ein XML- Dokument, welches fast denselben Regeln unterliegt wie andere XML- Dokumente. Es muss ebenfalls wohlgeformt sein und über ein Root- Element verfügen.

2. Aufbau der Dokumente

2.1. Aufbau von XML – Dokumenten

Auch wenn sich XML – Dokumente einer großen Freiheit bei den Möglichkeiten Ihrer Gestaltung erfreuen, gibt es einige feste Regeln, um deren Wohlgeformtheit und Gültigkeit sicher zu stellen.
Man unterscheidet hier zwischen dem physischen und dem logischen Aufbau eines XML – Dokumentes.

Zum physischen Aufbau zählen zumeist die ersten Zeilen eines solchen Dokuments, in welchen sogenannte Entitäten festgelegt werden. Die erste Entität ist grundsätzlich die Hauptdatei des XML – Dokumentes.

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

Weiter folgen Entitätsreferenzen, welche zum Beispiel auf die zu verwendende Dokumenttypdeklaration oder andere benötigte Dateien hinweisen.

In einigen Fällen enthalten XML – Dokumente eine interne Dokumenttypdeklaration, welche den logischen Aufbau des Dokumentes beschreibt.

Bei dem logischen Aufbau eines XML – Dokumentes handelt es sich um die bereits erwähnte Baumstruktur eines XML – Dokumentes, welches aus den folgenden Elementen bestehen kann:

- **Elemente**, welche physisch durch Tags ausgezeichnet werden.

Ein Element besteht aus einem Start- und End- Tag
oder aus einem Empty - Element – Tag.

```
<Praesentation> ... </Praesentation>          <!--Start- / End-Tag-->
<Thema/>                                      <!—Empty – Element – Tag>
```

- **Attribute**, welche einem Element weitere Eigenschaften zuweisen

Bestehend aus Schlüsselwort – Wert – Paaren
Attribut-Name = ‚Attribut – Wert’

- **Verarbeitungsanweisungen**
- **Kommentare**

Kommentare werden von XML-Parsern beim Auslesen des Dokumentes nicht beachtet.

```
<!--Ich bin ein Kommentar -->
```

- **Und Textabschnitte**

Welche den eigentlichen Inhalt des XML – Dokumentes darstellen.

2.2. Aufbau von Dokumenttypdefinition

Die Dokumenttypdefinition dient wie bereits erwähnt dazu, den logischen Aufbau von XML– Dokumenten zu beschreiben.

Folgende Elemente können in einer DTD definiert werden:

- Elementtypen
- Attributlisten
- Entities
- Notationen
- NMTOKEN
- Parameter-Entities
- Bedingte Abschnitte

Um die Validierung von XML – Dokumenten zu beschreiben wollen wir uns hier aber nur mit den Elementtypen beschäftigen.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT praesentation (Einleitung, Hauptteil, Abschluss)>
<!ELEMENT Einleitung (Begruessung, Definition*)>
<!ELEMENT Begruessung (#PCDATA)>
<!ELEMENT Definition (Stichwort, Deftext)>
<!ELEMENT Stichwort (#PCDATA)>
<!ELEMENT Deftext (#PCDATA)>
<!ELEMENT Hauptteil (Thema+)>
<!ELEMENT Thema (Titel, Thementext)>
<!ELEMENT Titel (#PCDATA)>
<!ELEMENT Thementext (#PCDATA)>
<!ELEMENT Text (#PCDATA)>
<!ELEMENT Abschluss (Fragen*, Verabschiedung)>
<!ELEMENT Fragen (Frage, Antwort)>
<!ELEMENT Frage (#PCDATA)>
<!ELEMENT Antwort (#PCDATA)>
<!ELEMENT Verabschiedung (#PCDATA)>
```

Wie in diesem Beispiel zu erkennen, werden in einer DTD die Elemente einzeln unter Angabe ihres Inhaltes, bzw. ihrer „Kind-Elemente“ definiert.
Im Bezug auf die Definition der Elemente ist die Festlegung der möglichen Anzahl von Kindelementen besonders im Hinblick auf die Validierung von Bedeutung.
Folgende Suffixe werden für diese Festlegungen verwendet:

? Kein oder ein Element ist erlaubt

* Kein oder mehrere Elemente sind erlaubt

+ Macht ein oder mehrere Elemente erforderlich

Wird kein Suffix verwendet, so ist genau ein Kind-Element erforderlich.

Weiter ist es möglich dem Inhalt eines Elementes eine Auswahl zuzuordnen:

```
<!Element Ziffer (eins|zwei|...|neun|null)>
```

2.3. Aufbau von XML - Schemata

Im Gegensatz zu DTD besteht der Hauptunterschied bei XML – Schema darin, dass die Definition mehrerer bestimmter Datentypen möglich ist.
Bei XML – Schema kann man zuerst eine Unterteilung zwischen „Complex Types“ und „Simple Types“ erkennen. Bei den „Simple Types“ handelt es sich um Typen, die die möglichen Inhalte von „Complex Types“ beschreiben. Hier gibt es eine Reihe vordefinierter Typen, die im XML – Schema zur Verfügung stehen.
Folgende Typen sind vordefiniert:
String, normalizedString, token, byte, unsignedByte, base64Binary, hexBinary, integer, positiveInteger, negativeInteger, nonNegativeInteger, nonPositiveInteger, int, unsignedInt, long, unsignedLong, short, unsignedShort, decimal, float, double, Boolean, time, dateTime, duration, gMonth, gYear, gYearMonth, gDay, gMonthDay, Name Lieferadresse, Qname, NCName, anyURI, language, ID, IDREF, IDREFS, ENTITY, ENTITIES, NOTATION, NMTOKEN, NMTOKENS

Von diesen einfachen Typen kann man nun auch eigene einfache Typen ableiten und somit zum Beispiel den Typ „Integer“ auf bestimmte Wertebereiche einschränken, oder beim Typen „String“ nur bestimmte Zeichen zulassen.

Ebenso wie bei einer DTD lassen sich natürlich auch bei einem XML - Schema „Elemente“ und „Attribute“ definieren. Hierbei handelt es sich um die „Complex Types“.

Auffällig an einem XML – Schema, ist das alle Elemente des Schemas mit einem Präfix „xs:“ bzw. „xsd:“ beginnen, welche über eine Deklaration (xmlns:xsd=’http://www.w3c.org/2001/xmlSchema“ am Beginn des Schemas die Verbindung mit dem XML – Schema Namensraum herstellen.

```
<!--XML – Schema, Beispiel von edition-w3c.de-->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element name="Bestellung" type="BestellungTyp"/>
<xsd:element name="Kommentar" type="xsd:string"/>
<xsd:complexType name="BestellungTyp">
  <xsd:sequence>
    <xsd:element name="Lieferadresse" type="DeAdresse"/>
    <xsd:element name="Rechnungsadresse" type="DeAdresse"/>
    <xsd:element ref="Kommentar" minOccurs="0"/>
    <xsd:element name="Waren" type="WarenTyp"/>
  </xsd:sequence>
```

3. Validierung

3.1. Wofür nun eine Validierung?

Einer der großen Vorteile von XML ist die Plattformunabhängigkeit. XML kann nicht nur für Webanwendungen sondern für eine Datenweitergabe in allen Bereichen der EDV auch zwischen unterschiedlichen Plattformen verwendet werden.

Doch gerade bei der Verwendung von unterschiedlichen Plattformen können Probleme auftreten, welche eine Validierung der Dokumente, zur Vermeidung von späteren Fehlern ausschließen.

Beide Partner, welche Daten mittels XML austauschen wollen, müssen sich nur auf einen einheitlichen logischen Aufbau der Dokumente einigen und können die Erstellung der Dokumente nun ihren eigenen Bedürfnissen und Mitteln anpassen. Um jedoch eine eventuelle Fehlerhaftigkeit der Dokumente frühzeitig aufzudecken, sollten diese von beiden Seiten validiert werden.

Die erfolgreiche Verwendung von XML bedarf also in diesem Falle nicht nur der Wohlgeformtheit, sondern auch der Gültigkeit. Die Gültigkeit kann dann festgestellt werden, wenn ein XML – Dokument mit Hilfe der zu Grunde liegenden Dokumenttypdefinition validiert, also auf Übereinstimmung geprüft wird.

Bei einfachen Webanwendungen hingegen ist die Validierung nicht erforderlich, da Browser nur die Wohlgeformtheit eines Dokumentes verlangen. Wenn hier Daten, die laut DTD zwingend erforderlich sind, fehlen, so wird der entsprechende Inhalt z.B. einfach nicht angezeigt, wodurch nur in seltenen Fällen ernste Fehler entstehen können.

Ein großer Vorteil der Verwendung von XML – Dokumenten liegt darin, dass ein breites Spektrum an XML – Parsern auch zur freien Verwendung zur Verfügung steht, welche die Validierung von XML – Dokumenten übernehmen. So ist es bei der

Softwareentwicklung nicht nötig eigene Dateiformate zu entwickeln und aufwendige Routinen zur Prüfung der Gültigkeit solcher Formatdateien zu entwickeln. Somit entstehen auch bei der Programmierung weniger Fehler und neue Programme können schneller zum Einsatz kommen.

3.2. Was wird validiert?

Es werden sowohl die physische Struktur (wobei man hier eher von Wohlgeformtheit) als auch die logische Struktur bzw. der Inhalt eines XML – Dokumentes validiert. Auch hier sollen einige Beispiele die Bedeutung verdeutlichen, da ein Eingehen auf alle nur möglichen Fehler die in XML – Dokumenten auftreten können den Rahmen sprengen würden.

Feststellung der Wohlgeformtheit

Validierung der logischen Struktur

Validierung des Inhaltes

=>Und somit Feststellung der Gültigkeit eines Formulars.

Hierzu einige Beispiele: (zugehörige DTD siehe Seite 9)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE praesentation SYSTEM "C:\Altova Projects\Praesentation.dtd">
<praesentation>
        <Einleitung>
```

Gültiger Beginn eines XML – Dokumentes

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

<praesentation>
        <Einleitung>
```

Ungültiger Beginn eines XML – Dokumentes. Es wird auf keine DTD und kein Schema verwiesen, gegen welches das XML – Dokument validiert werden kann.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE praesentation SYSTEM "C:\Altova Projects\Praesentation.dtd">
<praesentation>
        <Hauptteil>
                <Thema>
                        <Titel>Aufbau von XML - Dokumenten</Titel>
                        <Thementext></Thementext>
                </Thema>
                <Thema>
```

Ungültiger Beginn des Dokumentes, da die DTD eine Einleitung vor dem Beginn des Hauptteils vorschreibt.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE praesentation SYSTEM "C:\Altova Projects\Praesentation.dtd">
<praesentation>
        <Einleitung>
                <Literaturverweis>Buch</Literaturverweis>
                <Definition>
                        <Stichwort>XML</Stichwort>
                        <Deftext></Deftext>
```

Ungültig, da das Element Literaturverweis in der DTD nicht vorgesehen ist.

4. Praxisteil

Um das ganze noch einmal ganz praktisch zu sehen, wollen wir uns das anhand eines Beispiels verdeutlichen. Hierzu stellen wir uns folgende Aufgabenstellung vor:
Wir befinden uns in einer Bibliothek für die ein Literaturverzeichnis zu erstellen ist. In diesem werden Bücher aufgenommen. Jedes Buch hat einen Autor, einen Titel, einen Verlag und ein Erscheinungsjahr. Des Weiteren ist bekannt, dass alle Bücher entweder in deutsch, englisch, spanisch oder französisch vorliegen. Eine letzte Besonderheit ist, dass die meisten Bücher (Aufgrund reger studentischer Nachfrage) aus dem Vieweg-Verlag und konkreter noch aus der Feder Lothar Papulas stammen.

Daraus würde sich folgende Struktur ergeben:

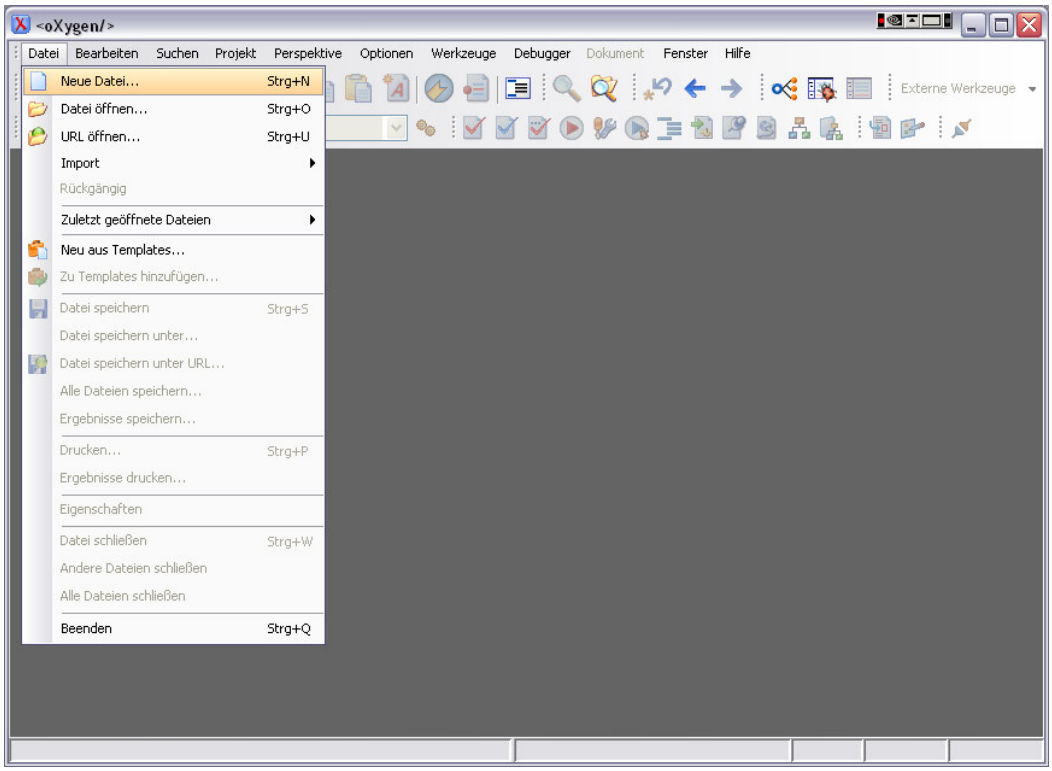
Literaturverzeichnis

- Buch (Autor, Titel, Verlag, Datum)
    - Sprache (deutsch|englisch|spanisch|französich)
- Buch (Autor, Titel, Verlag, Datum)
    - Sprache (deutsch|englisch|spanisch|französich)
- Buch (Autor, Titel, Verlag, Datum)
    - Sprache (deutsch|englisch|spanisch|französich)
- Buch…

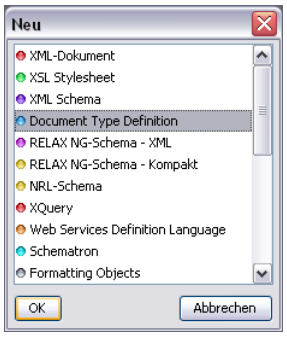
Anmerkung:
Es werden selbstverständlich mehrere Bücher aufgenommen, jedoch in nur ein Literaturverzeichnis!

Hierfür erstellen wir zuerst eine DTD. Das tun wir mit dem Programm <oXygen/>, welches für eine Testzeit von 30 Tagen kostenfrei unter http://www.oxygenxml.com/ herunter geladen werden kann.

Wir öffnen das Programm und wählen unter <Datei> Neue Datei…

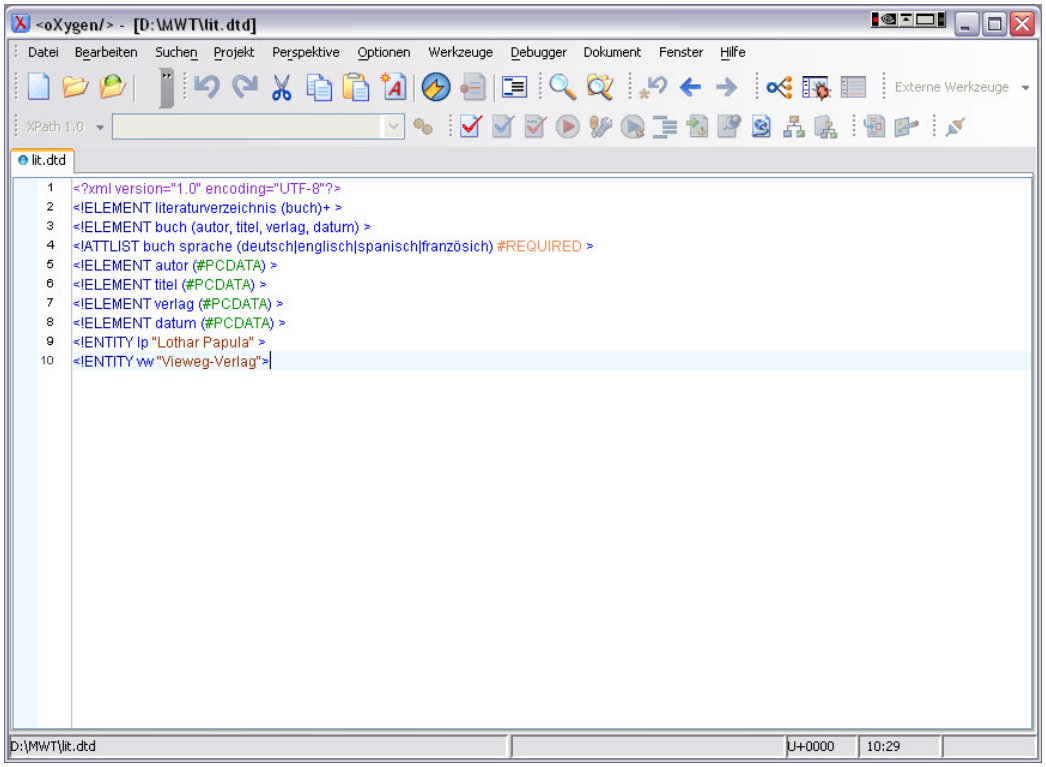


In der nachfolgenden Auswahl wählen wir „Document Type Definition“

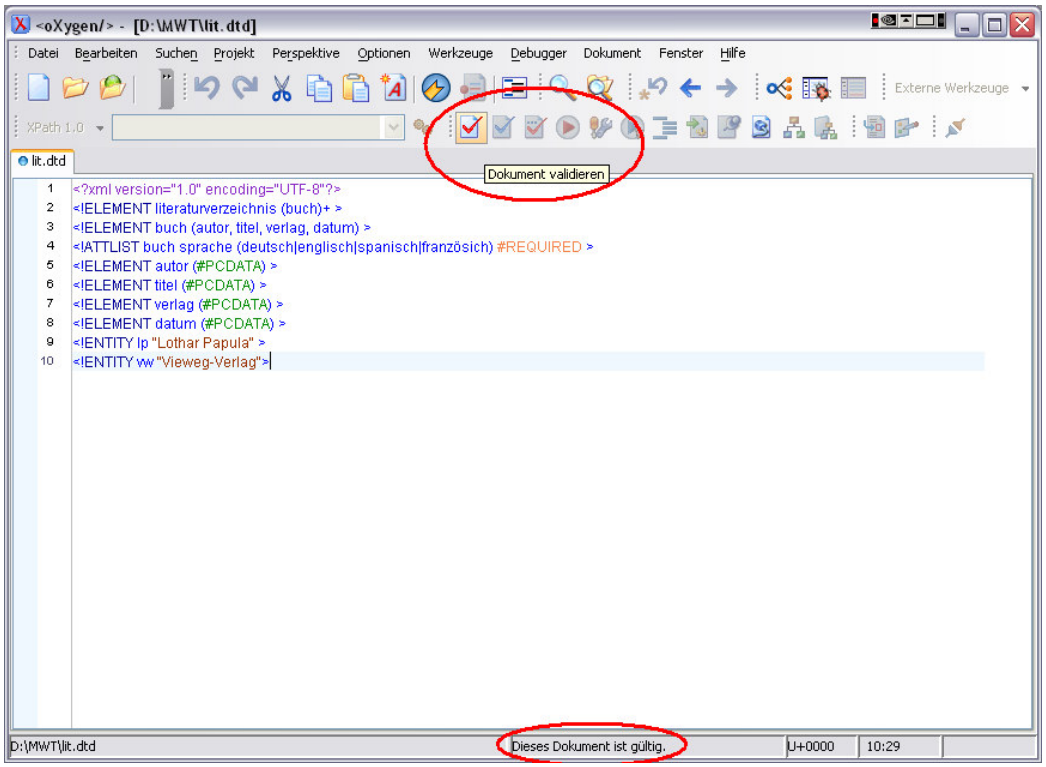


Und bestätigen dieses.

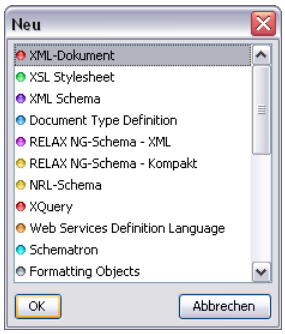
Nun können wir den Code für die DTD eingeben und speichern diesen unter lit.dtd ab.



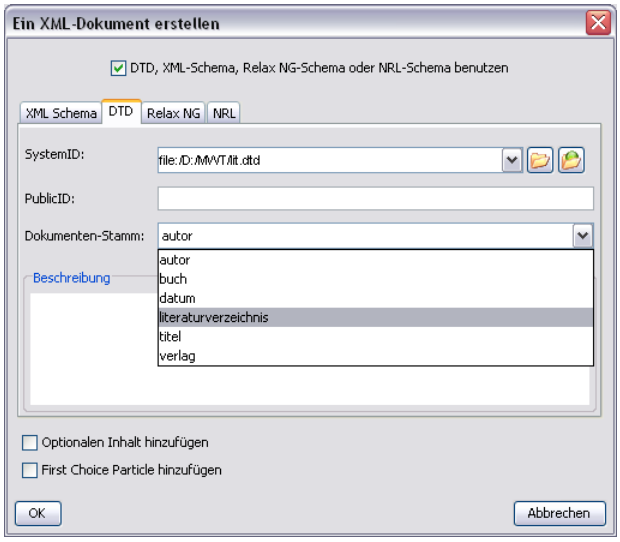
Bevor wir mit der Dateneingabe fortfahren sollten wir überprüfen, ob unsere DTD gültig ist und klicken auf „Dokument validieren“, worauf in der Statusleiste das Ergebnis erscheint: „Dieses Dokument ist gültig.“



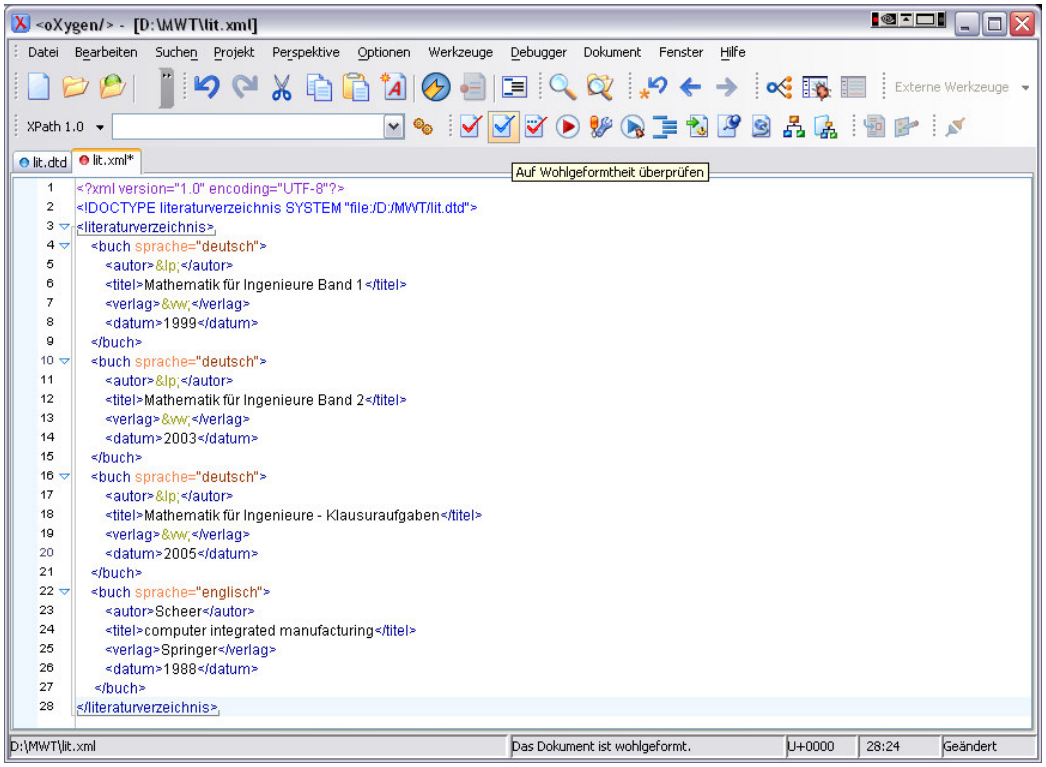
Soweit so gut, kommen wir zum eigentlichen XML-Dokument. Dazu müssen wir wieder ein neues Dokument erstellen. Diesmal jedoch ein „XML-Dokument“ welches wir nach Auswahl bestätigen.



Es erscheint ein weiters Auswahlfenster, in welchem wir in der Kartei „DTD“ unsere abgespeicherte „lit.dtd“ aufgerufen wird und im Dokumenten-Stamm das „literaturverzeichnis“ angewählt wird.

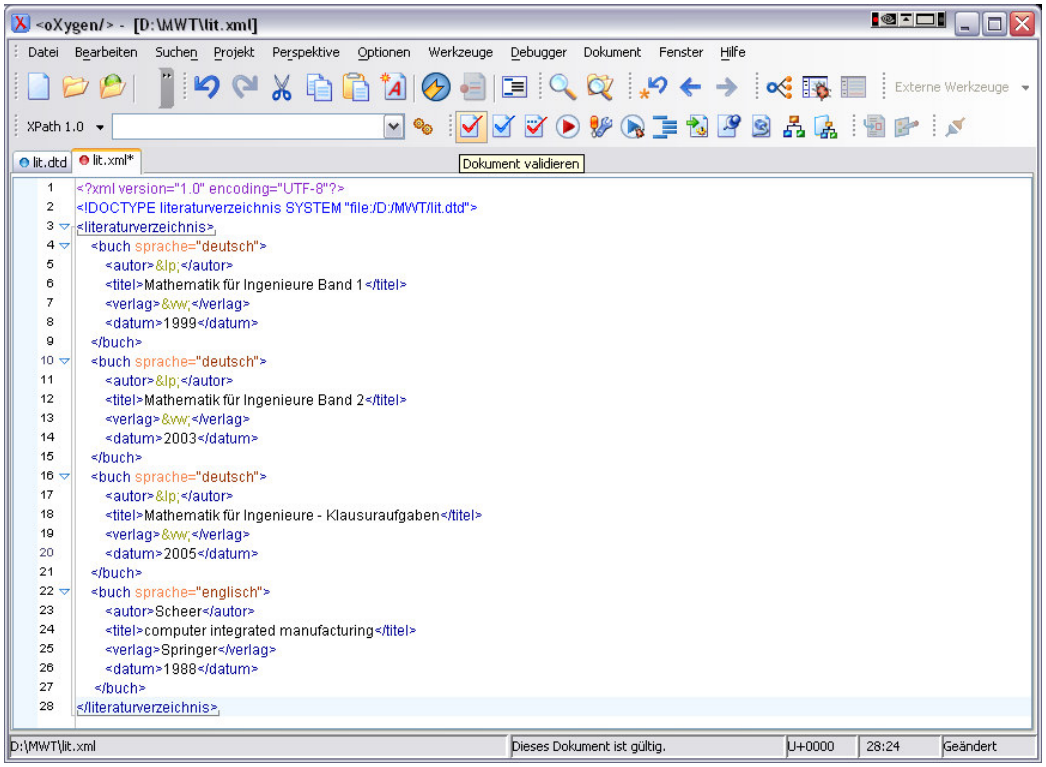


Nach Bestätigung auf „OK“ wird nun eine vordefinierte XML-Datei erzeugt, die wir unter „lit.xml“ abspeichern.
Nach einigen Änderungen und Eintragungen können wir unser Dokument auf die Wohlgeformtheit überprüfen.

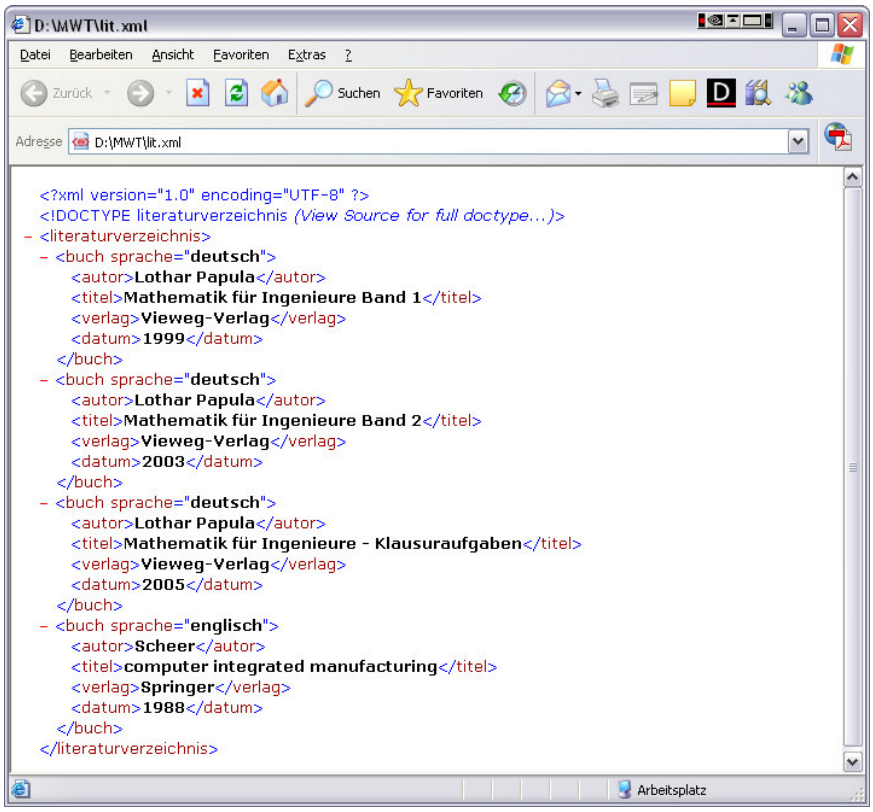


Ist das XML-Dokument wohlgeformt (wie man in der Statusleiste sehen kann), können wir es auch anhand einer DTD validieren.

Da die DTD im <!DOCTYPE> festgelegt ist, brauchen wir diese nicht noch erst zuweisen, sondern klicken auf „Dokument validieren“.



Wie erwartet ist auch dieses Dokument gültig und wir können einen Blick in den Internetexplorer wagen.



Hier können wir jetzt erkennen, dass wirklich alles bestens funktioniert, denn sowohl der Autor, als auch der Verlag wurde richtig ersetzt. Wir stellen ebenfalls fest, dass wir nur ein Wurzelelement haben und entdecken nebenbei, dass beim klicken auf ein (-) sich der ganze Ast schließt.

5. Quellenangaben

- http://www.edition-w3c.de

  Die W3C-Spezifikationen in deutscher Übersetzung und Kommentierung
- http://www.w3c.org
- XML in a nutshell

  Elliotte Rusty Harold & W. Scott Means

  O'REILLY Verlag Köln
- Workshop XML

  Magnus Stein

  Addison – Wesley Verlag
- Einführung in XML

  Erik T. Ray

  O'REILLY Verlag
- http://www.wikipedia.de
- SELFHTML

  http://de.selfHTML.org