OVIDIUS

DITA - eine neue Strukturierungsmethode?

Vortrag auf dem Doku+Medien Forum 2006

OVIDIUS

Vorwort

Selten hat ein XML-basiertes Konzept zur Erfassung und Publikation technischer Informationen in so kurzer Zeit so viel Aufsehen und Aufmerksamkeit erregt wie DITA. Es soll daher betrachtet werden, worum es sich dabei handelt und wie DITA in der technischen Dokumentation eingesetzt werden kann. Nachdem der erste Hype und die damit einhergehende Euphorie vorüber ist, ist es an der Zeit zu einer von weniger religiösem Eifer begleiteten Betrachtung. Nach einer kurzen Einführung in die Philosophie von DITA soll untersucht werden, wie DITA nutzbar gemacht werden kann. Ebenso sollen die Grenzen von DITA aufgezeigt werden.

Torsten Machert

Ovidius GmbH

Inhaltsverzeichnis

1 Die Philosophie

DITA - eine neue Struktuerierungsmethode?

Diese Frage kann ohne viel Nachdenken verneint werden. Während beispielsweise das Funktionsdesign und InformationMapping® "richtige" Strukturierungsmethoden sind, die prinzipiell auch ohne die Verwendung von XML funktionieren, handelt es sich bei DITA um einen Zwitter, der sowohl eine Architektur als auch eine DTD ist. Der Begriff *DITA* steht für *Darwin Information Typing Architecture*.

Dem Begriff kann man entnehmen, dass es sich um eine Architektur handelt, in der es um typisierte Informationen geht. Für bestimmte Informationsarten soll es spezialisierte Container geben. Wir werden im folgenden auf die unterschiedlichen Informationsarten eingehen. Häufig wird das Wort *Darwin* immer dann verwendet, wenn man meint, etwas in den Zusammenhang mit Vererbung stellen zu müssen. Bei Darwin geht es jedoch in erster Linie nicht um Vererbung sondern um Spezialisierung und Anpassung. Genau darum geht es auch bei DITA: Auf der Grundlage einer durch eine Basis-DTD ausgedrückten Basis-Architektur soll es möglich sein, eigene Informationsbedürfnisse abbilden zu können. Wichtig ist dabei, eine Rückwärtskompatibilität zu gewährleisten: Jede Struktur muß sich auf die Ausgangsstruktur abbilden lassen.

DITA ist ein XML-basiertes Konzept. Das soll deshalb betont werden, weil einige Hersteller Microsoft Word®-basierter Redaktionssysteme nicht müde werden zu verkünden, DITA zu unterstützen. Dass sich aus einem entsprechend strukturierten Worddokument ein XML-Dokument erzeugen läßt, dass der DITA-Basis-DTD genügt, soll nicht in Frage gestellt werden. Die weiter unten besprochenen Mechnismen zur Spezialisierung lassen sich mit Nicht-XML-Mechanismen nicht abbilden. Der Architekturgedanke von DITA ist mit proprietären Dateiformaten nicht abbildbar.

Alles neu?

Nichts an DITA ist wirklich neu. Das ist aber auch keine Schande. Aus Erfahrungen zu lernen, um modifizierte Konzepte zu entwickeln, ist ein probater Weg. Nach eigener Aussage ist DITA was die Art der Informationserstellung und vor allem aber was die Art der Informationsrezeption betrifft vom InformationMapping® beeinflußt. Auch die bereits erwähnte Möglichkeit der Erweiterung und Spezialisierung der Basis-DTD stellt keine Neuheit dar. Bereits in den 1990er Jahren wurde mit den Architectural Forms ein Mechanismus diskutiert und letztlich zum ISO-Standard erhoben, der sich auch für das Erstellen von DTD-Ableitungen eignet.

Der für DITA typische Topic-zentrierte Ansatz findet in modifizierter und ungleich mächtigerer Form bereits seit vielen Jahren in der europäischen Verteidigungsindustrie in Form der S1000D-Spezifikation eine umfangreiche Anwendung.

Ein Standard?

Der Begriff *Standard* wird häufig auch für Dinge verwendet, die man Quasi- und/oder Industriestandard nennen würde. DITA ist definitiv kein Standard, wenn man damit Dinge meint, die von internationalen oder nationalen Standardisierungsorganisationen als solche betreut und veröffentlicht werden. Wenn man mit dem Wort *Standard* einen hohen Verbreitungsgrad signalisieren will, ist DITA auch (noch?) kein Standard.

Das Streben nach Standards drückt häufig auch das Bemühen aus, bei Entscheidungen keinen Fehler zu machen. Nach dem Motto: Wenn es so viele andere und so viele Große einsetzen, dann muß es ja auch gut und richtig sein. Ein wichtiger Grund für das Verwenden von Standards in bestimmten Industrien

ist die Austauschbarkeit von Daten. Welcher Hersteller hat nicht das Interesse, die von seinen Lieferanten kommenden Dokumentationsbestandteile nahtlos in seine Dokumentation einzubinden?

Solche Quasistandards gibt es in bestimmten Industrien bereits. Zu nennen wären die Automobil-, Luftfahrt-, Verteidigungsindustrie sowie die pharmazeutische Industrie. Sie haben sich Strukturen geschaffen, die ihnen einerseits den Datenaustausch ermöglichen. Andererseits haben die erwähnten industriespezifischen Standards eine stark auf die jeweilige Industrie zugeschnittene Semantik. Man könnte zwar eine Zulassungsdokumentation aus der Pharmazie oder einen Beipackzettel auch mit den Mitteln von DITA erstellen. Wegen der fehlenden Spezialisierung und der hohen Beliebigkeit der DITA-Strukturen wird man es jedoch idealerweise nicht tun. Gerade in Industrien wo eine sehr große Anzahl von Personen Informationen in Form strukturierter Daten liefern muß, wird man eher rigide Strukturen verwenden, um eine einheitliche und konsistente Strukturierung zu garantieren.

Ein Flughandbuch soll einem Piloten gerade in Notfällen helfen, gleiche Informationen stets in gleicher Reihenfolge zu finden. Eine geeignete XML-DTD hilft den Autoren, die Informationen entsprechend zu erfassen.

Buch vs. Topic

Neben dem allbekannten Docbook ist DITA das zweite XML-basierte Dokumentationskonzept, das durch die OASIS (Organsation for the Advancement of Structured Information Standards) betreut wird. Es stellt sich die Frage, was neu und konzeptionell anders als in Docbook ist bzw. warum Docbook nicht ausreichend ist.

Docbook folgt, wie es der Name schon vermuten läßt, dem Buchparadigma. Autoren werden angehalten, in Buchkategorien zu denken und zu schreiben. Durch eine semantische Struktur, die die Hierarchietiefe vorgibt (chapter, sect1, sect2 et.) besteht keine Möglichkeit, Informationen in höheren oder tieferen Hierarchieebenen zu verwenden. Für Autoren ist der Kontext stets erkennbar, in dem eine Information publiziert wird.

DITA kennt kein Buch im eigentlichen Sinne des Wortes. Autoren schreiben keine Handbücher. Autoren schreiben Informationshäppchen, die z.B. geräte- oder zielgruppenspezifisch zu Publikationen zusammen gestellt werden. Handbücher stellen somit *eine* Sicht auf den Informationsbestand dar. Das Bedürfnis, Informationen so zu erfassen, ergibt sich aus den Bedürfnissen einer kontextsensitiven Hilfe für Softwareprodukte. Benutzer erwarten genau die Information, die im jeweiligen Kontext relevant ist. Die sich ihnen öffnende Hilfeinformation soll auch in sich abgeschlossen sein. Das Hilfethema zum Thema "Speichern" soll also beispielsweise nicht in das Thema "Drucken" übergehen. Informationen, die nach DITA geschrieben werden, sind das, was man im englischen "self-contained" nennt. Es sind abgeschlossene, kontextfrei funktionierende Informationen. In der DITA-Welt werden diese Informationshäppchen *Topic* genannt. Wir werden unten sehen, welche Ausprägungen ein Topic haben kann.

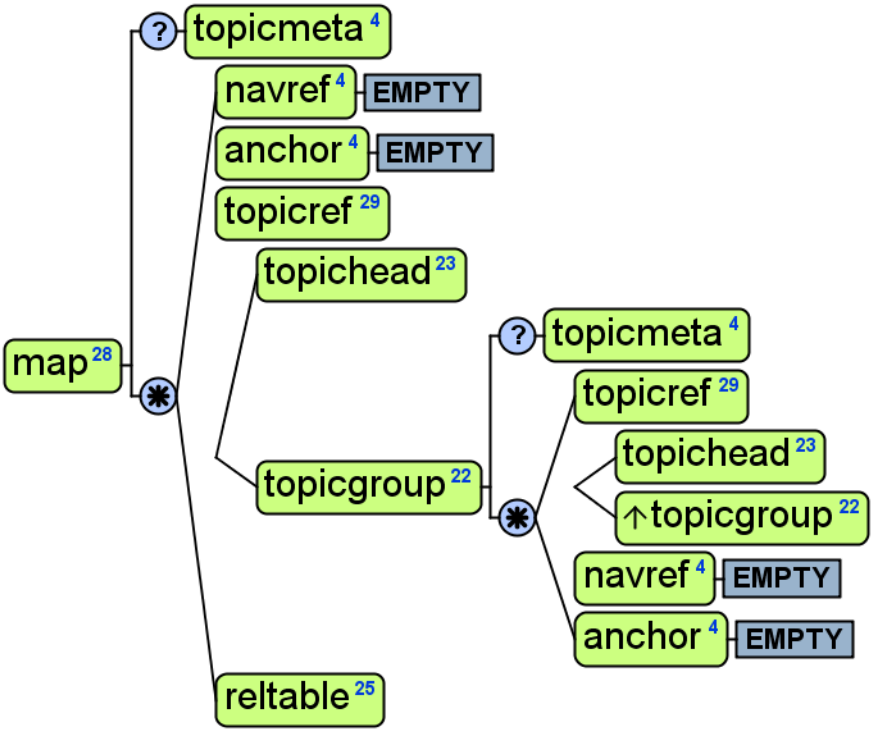
Eine solche Art der Informationserfassung setzt eine umfassende Planung des redaktionellen Prozesses voraus. Das sollte zwar auch für buchähnliche Dokumentationen gelten. Sie "verzeihen" jedoch schon eher Versäumnisse, weil fehlende Informationen irgendwie "hineingefummelt" werden können. In der Welt, aus der DITA kommt, in der Softwaredokumentation basiert ein großer Teil der Planung auf der Interaktion zwischen der Redaktion und der Entwicklungsabteilung und/oder dem Produktmanagement. Wenn man einmal den Idealfall unterstellt, dass der kontextsensitiven Hilfe ein Konzept zugrunde liegt, dann ist klar, dass es für jede durch die SW-Entwickler vergebene Hilfe-ID ein Topic geben muß.

Ein anderer Weg der Planung könnte darin bestehen, dass man so verfährt, wie man es in der Verteidigungsindustrie tut, wo nach der ASD S1000D dokumentiert wird: Ausgehend von einem Systemaufbruch wird pro System, Subsystem, Komponente definiert was dokumentiert werden muß. Wenn wir annehmen, dass zu einer Komponente A folgende Informationen dem Kunden/Nutzer an die Hand gegeben werden müssen: Aufbau, Bedienung, Einbau, Ausbau, Schmierung, dann benötigt man für jede dieser Informationen ein Topic. Aus der Art der zu veröffentlichenden Informationen ergibt sich dann auch die "Größe" eines Topic. Die häufig gestellte Frage nach der Größe eines Topic kann also nicht allein nach formalen Kriterien beantwortet werden.

Der oben beschriebene Topic-basierte Ansatz von DITA ermöglicht, wenn eine sorgfältige Planung erfolgt ist, das Generieren nahezu beliebiger Sichten auf den Informationsbestand. Wenn man es etwas weniger akademisch ausdrückt, kann man auch sagen, dass nahezu beliebige Informationsprodukte wie z.B Benutzerhandbücher, Administrator- oder Referenzhandbücher zusammen gestellt werden können.

Um sicherzustellen, dass solche durch die Komposition einer großen Zahl von Topics entstehenden Dokumentationsprodukte "lesbar" sind und bleiben, muß der redaktionelle Prozess durch einen geeigneten Redaktionsleitfaden und eine gute Schulung aller Autoren begleitet werden. Den Topics soll nicht unbedingt angemerkt werden, welcher Autor dafür verantwortlich zeichnete. Schreibstil, Formulierungsmuster, die Verwendung von Terminologie usw. müssen vorgegeben werden und ausführlich geschult werden. Ein Lektorat wird durch die Verwendung von DITA ebenfalls nicht obsolet.

Für das XML-basierte Zusammenstellen der verschiedenen Topics stellt DITA eine so genanante `Map` zur Verfügung.



Die Struktur von map

Für eine umfassende Erläuterung verweise ich wiederum auf die DITA-Dokumentation. An dieser Stelle soll lediglich das Prinzip erläutert werden. Die *map* beschreibt den Inhalt einer Dokumentation und die hierarchische Anordnung

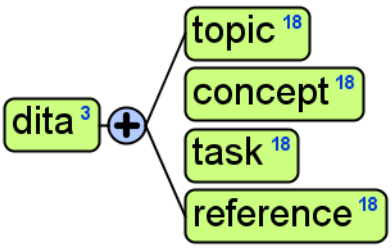
der Komponenten. Neben Titel- und Navigationselementen sowie Strukturen zur Erfassung von Metainformationen gibt es das Element `topicref`, dass ein Topic über sein Attribut `href` referenziert. Da es eine rekursive Struktur aufweist, lassen sich auch Hierarchien aufbauen.

Für eine Online-Version einer Dokumentation ist dieser Mechanismus gut geeignet. Prinzipiell kann man aus einer solchen *map* auch ein gedrucktes Werk erstellen. Allerdings ist in Abhängigkeit vom verwendeten Werkzeug unter Umständen ein Vorprozess notwendig, der die referenzierten Topics in das Produkt kopiert.

Die Architektur

DITA will mehr sein als nur eine weitere DTD für das Erfassen und Publizieren von technischen Informationen. Es soll an neue Informationsbedürfnisse anpassbar sein. Da im Kontext von XML alle Publikationsprozesse immer DTD-basiert und DTD-spezifisch sind, führen Modifikationen und Ergänzungen von DTDs stets zu entsprechenden Änderungen der Publikationsprozesse.

Zum besseren Verständnis werden in den Text Abbildungen von DTD-Bereichen eingestreut. Begonnen werden soll, mit der obersten DITA-Struktur.



DITA-Struktur

Die DITA-Idee der Anpassung bzw. Spezialisierung ist, dass neue Elemente als Spezialisierung vorhandener Elemente definiert werden. Betrachten wir zur Verdeutlichung die in der ditabase-DTD definierten Elemente `Concept`, `Reference` und `Task`. Sie stellen bereits Spezialisierungen der Mutter aller DITA-Elemente `Topic` dar.

In der ditabase-DTD finden folgende Attributedefinition für das Element `Concept`:

```
<!ATTLIST concept      %global-atts;
class CDATA "- topic/topic">
```

Der Schlüssel zum Verständnis steckt in der Definition des Attributs `class`. Dieses Attribut wird verwendet, um zu definieren, auf welche Weise das jeweilige Element durch Spezialisierung entstanden ist. Das Minuszeichen bedeutet, dass es sich um eine strukturelle Spezialisierung handelt. Ein Pluszeichen würde bedeuten, dass es sich um eine domänspezifische Spezialisierung handelt. Der Ausdruck `topic/topic` bedeutet, dass das Element `concept` eine Spezialisierung von `topic` ist.

Diese Art der Spezialisierung hat den Vorteil, dass Erweiterungen bzw. Spezialisierungen einer bestehenden Struktur nicht unbedingt zu Modifikationen der verwendeten Publikationslösungen führen müssen. Das setzt jedoch voraus, dass alle Transformationen und Formatierungen nicht primär auf Elementregeln basieren sondern auf Attributregeln. Im Beispiel oben haben wir es zwar mit dem Element `concept` zu tun. Es könnte aber durch die Berücksichtigung des Wertes des `class`-Attributs bei der Publikation so behandelt werden wie das Element `topic`.

Wie die Autoren von DITA betonen, gibt es Beschränkungen bei der Spezialisierung:

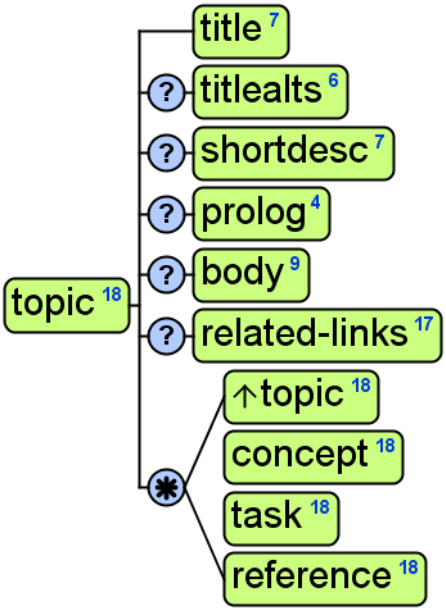
- Wenn es den Bedarf nach Informationsstrukturen gibt, die sich nicht aus bestehenden DITA-Strukturen ableiten lassen, muß man die Finger von DITA lassen oder einen DITA-inkompatiblen Weg gehen
- Durch Spezialisierung entstandene Elemente müssen das gleiche oder ein restriktiveres Inhaltsmodell als das Element haben, das spezialisiert wurde
- Die Attribute des durch Spezialisierung entstandenen Elements müssen mindestens denen entsprechen, die im Ursprungselement vorhanden sind. Ebenso müssen die Wertebereiche eines Attributs in beiden Element identisch sein.

Der Spezialisierungsmechanismus von DITA ist proprietär! Er beruht vollständig auf Verabredung, da er durch kein Werkzeug unterstützt wird. Um die Spezialisierung von DITA robust nutzen zu können, werden spezialisierte XML-Parser benötigt, die die Plausibilität der Spezialisierung bezogen auf die Spezialisierungsregeln überprüfen.

2 Die Realisierung

Die DTD

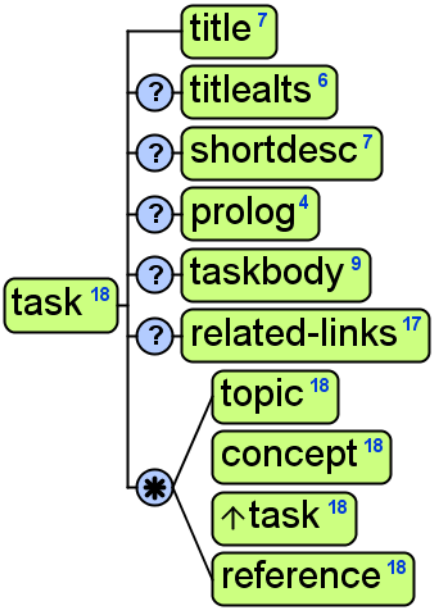
Wenn das Element `topic` die Mutter aller strukturbildenden Element ist, dann ist es interessant, einen Blick auf die Struktur dieses Elements zu werfen.



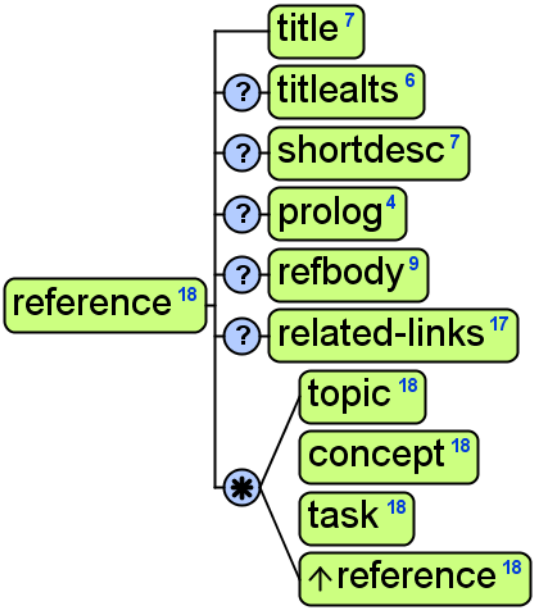
Struktur des topic-Elements

Hieran gibt es wenig auszusetzen: Einem obligatorsichen Titel folgen optionale alternative Titel wie sie in Navigationsstrukturen oder Suchanwendungen verwendet werden könnten. Die optionale Kurzbeschreibung und ein ebenfalls optionaler Prolog geben dem Autor Gelegenheit, den Leser/Nutzer angemessen in ein Thema einzuführen. Der eigentliche Inhalt des `topic` verbirgt sich dann in `body`, den wir in Kürze betrachten werden. Das darauf folgende Element `related-topics` verrät dann wieder das urpsüngliche Ansinnen von DITA, gute Hilfesysteme zu erstellen.

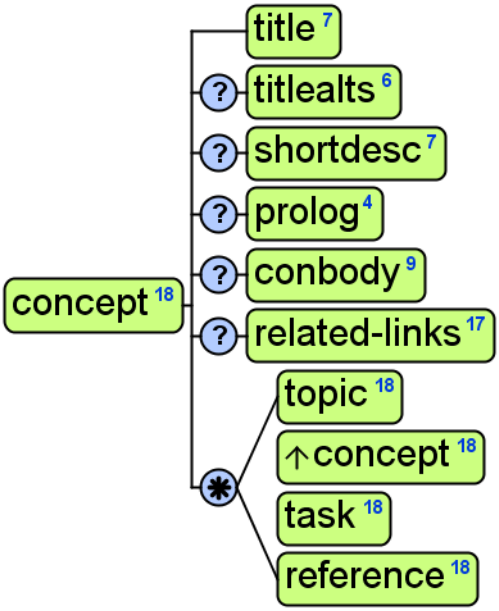
Diese Struktur findet sich dann auch in den von `topic` abgeleiteten Strukturen.



Struktur des task-Elements



Struktur des reference-Elements



Struktur des concept-Elements

task ist der Container für handlungsanleitende Informationen. Für beschreibende Informationen steht *concept* zur Verfügung. Für Referenzinformationen wie man sie häufig in Programmierhandbüchern findet, steht das Element *reference* zur Verfügung.

3 DITA in Praxis

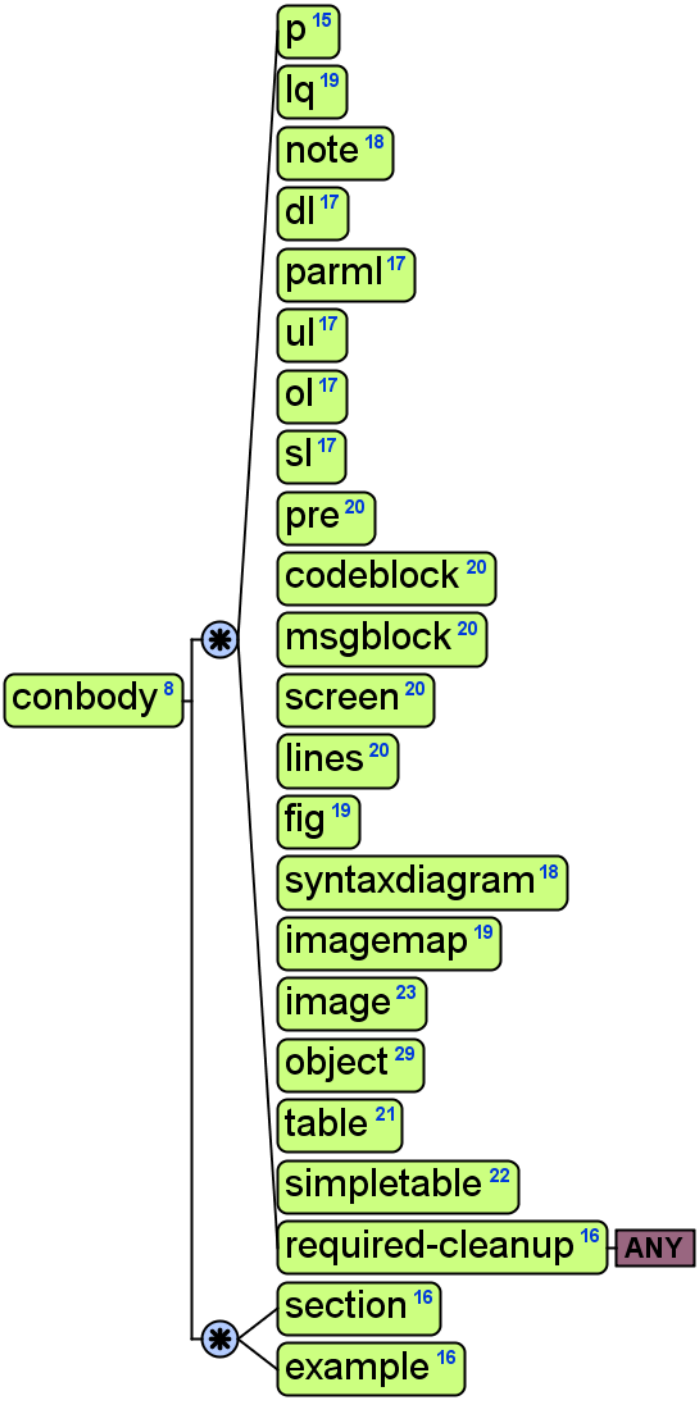
Hat man sich dafür entschieden, DITA für die eigene Dokumentation zu verwenden, stellt sich die Frage, ob die ditabase-DTD angemessen und ausreichend ist oder ob Modifikationen notwendig sind. Um eine Hilfestellung bei der Beantwortung dieser Frage zu geben, sollen einige Aspekte der ditabase-DTD detaillierter diskutiert werden. An dieser Stelle soll nicht die komplette ditabase-DTD betrachtet werden. Es wird lediglich auf ein paar Aspekte eingegangen, die helfen sollen, dem Leser eine eigene Betrachtung und Beurteilung zu ermöglichen.

Ein wichtiger Grund, XML in der technischen Dokumentation zu verwenden, besteht darin, dass es mit XML möglich ist, gleiche Informationsarten gleich zu strukturieren und sie damit unabhängig vom Verfasser konsistent zu halten. Es soll daher ebenso betrachtet werden, inwieweit die ditabase-DTD dieses Ziel unterstützt.

Mit XML soll es für Autoren einfacher werden, rechtssichere Dokumentationen zu schreiben. Eine gute und angemessene DTD wird Autoren durch eine geeignete semantische Struktur beim Schreiben entsprechend führen.

Beschreibende Informationen

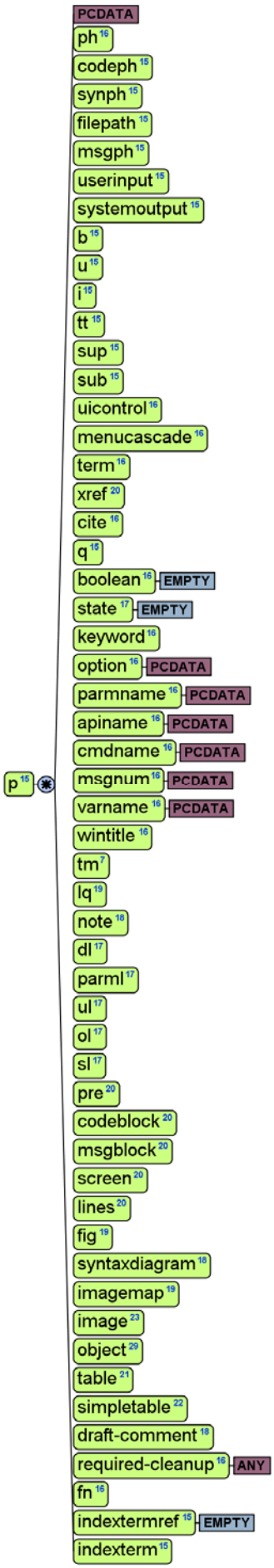
Wie wir oben gesehen haben, wird das Element *concept* für beschreibende Informationen verwendet.



Struktur von conbody

conbody stellt erwartungsgemäß eine Reihe von weiteren Informationsobjekten wie Absätzen, Tabellen, Listen, Abbildungen zur Verfügung, die man benötigt, um Texte zu strukturieren. Darüber hinaus gibt es ein paar für die Dokumentation von Software notwendigen Strukturen wie `screen` (Screenshots), `lines` (Codebeispiele) etc. zur Verfügung.

Die Anzahl der Element ist überschaubar. Sie sind mit vernünftigem Aufwand in einer Autorenschulung vermittelbar. Problematischer wird es, wenn wir einen Blick auf die Struktur von *p* werfen, das als allgemeines Absatzelement zur Verfügung steht.



Struktur von p

Es ist an dieser Stelle nicht notwendig, die Bedeutung aller Element zu besprechen. Allein die schiere Anzahl der Elemente läßt ahnen, wie hoch der Schulungsaufwand ist, um eine konsistente Struktur durch alle am Prozess beteiligten Autoren zu erreichen. Wenn wir schon ein Element haben, dass die Bezeichnung eines Fenstertitels (`wintitle`) haben, dann wollen wir auch, dass alle Autoren dieses Element zweckgebunden werden und nicht etwa das Element `i`, dass unter Umständen die gleiche Formatierung aufweist.

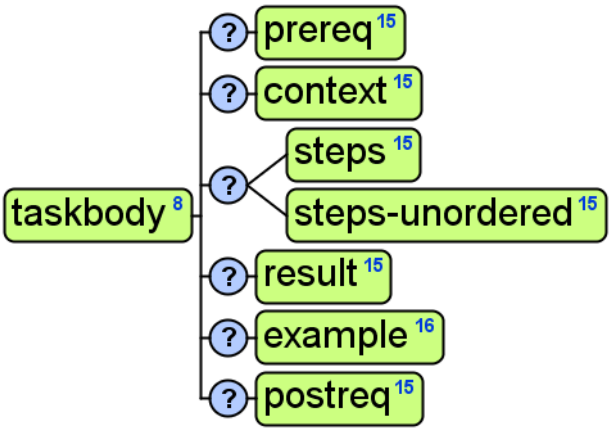
Eine mögliche Spezialiserung könnte hier darin bestehen, dass die Struktur ausgedünnt wird.

Wenn man sagt, dass `p` ein absatzartiges bzw. blockbildendes Format ist, dann ist zumindestens zweifelhaft, ob es noch als logisch betrachtet werden kannn, dass ein Blockelement andere Blockelemente wie Tabellen, Listen usw. als Kinder haben kann. Üblicherweise gelten solche Strukturen als schlechtes DTD-Design.

Trotz der großen Menge an Elementen wird auch hier deutlich, dass DITA auf die Bedürfnisse der Softwaredokumentation zugeschnitten ist. Da Maschinenbauer benötigen im Gegensatz dazu Elemente, die Ersatzteile, Zubehör, Schmierstoffe und Verbrauchsmaterialien auszeichnen. Die pharamazeutische Industrie benötigt wiederum ganz andere semantische Strukturen und Elemente. Will man in diesen Industrien nicht nur allgemeine Elemente `u`, `b` usw. sondern semantische Elemente verwenden, muß versucht werden, durch Spezialisierung zu den gewünschten Elementen zu kommen. Die Betonung liegt hier bewußt auf dem Wort *versuchen*. Damit die Struktur DITA-konform bleibt, sind die oben geschilderten Bedingungen einzuhalten.

Handlungsanleitende Strukturen

Für handlungsanleitende Strukturen steht das Element `task` zur Verfügung. Die eigentliche Handlungsfolge wird im Element `taskbody` beschrieben.

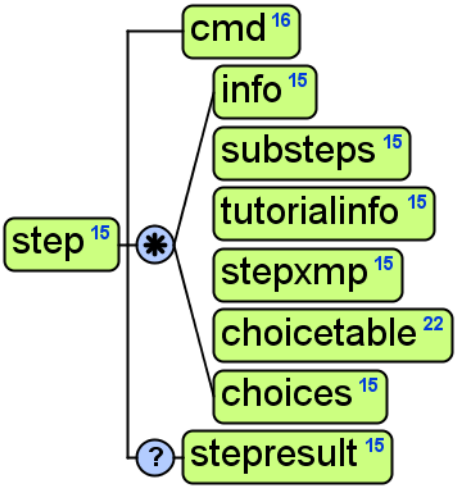


Struktur von taskbody

Bis auf die Tatsache, dass es ein Element `steps-unordered` gibt, sieht die Struktur plausibel aus: Nach den Voraussetzungen für eine Handlung (`prereq`) folgt die optionale Beschreibung des Kontextes, in dem eine Handlung sinnvoll und möglich ist. Neben dem bereits erwähnten Element `steps-unordered` gibt es auch das Element `steps`. Da sich der Autor zwischen sortierten und nichtsortierten (was immer das sein mag) Schritten zu entscheiden hat, ergibt sich daraus die Möglichkeit, dass in einer Publikation ein leerer taskbody erscheint, da aller Kinder dieses Elements optional sind.

Den Handlungsschritten folgen optional eine Resultatsangabe, ein Beispiel und Bedingungen, die nach der Handlung erfüllt sein müssen.

Wenn man von einer rechtssicheren Dokumentation spricht, dann meint man insbesondere die handlungsanleitenden Teile. Hier gibt es bestimmte Erwartungen und Vorgaben. Mit dem Mitteln von XML haben wir die Möglichkeit, Strukturen zu schaffen, die einem Autor helfen, rechtssicher und fachlich korrekt zu schreiben. Welche Hilfe bekommt man hierbei durch die ditabase-DTD? Betrachten wir dazu die Struktur des Element `step`, das Kind von `steps` und `steps-unordered` ist.



Struktur von step

Die eigentliche Anweisung steckt im Element `cmd`. Es fällt sofort auf, dass die ditabase-DTD weder auf der Ebene von `taskbody` noch auf der Ebene von `step` Strukturen hat, die für Warn- und Sicherheitshinweise vorgesehen ist. Das ist ein schwerwiegender Mangel. der DITA für das Gros der technischen Dokumentation unbrauchbar macht. Selbst die Argumentation, dass man so etwas in der Softwaredokumentation nicht benötige, verfängt nicht wie man an einem simplen Beispiel sehen kann: Dass das irrtümliche Formatieren eines Textes andere Konsequenzen hat als das irrtümliche Formatieren einer Festplatte mag durch den gesunden Menschenverstand erkennbar sein. Angesichts der Klagefreude in einigen Teilen der Welt tut man als Softwarehersteller gut daran, den Benutzer vor den Konsequenzen des Befehls `format c:` zu warnen.

Es ist jedoch zweifelhaft, ob man durch Spezialisierung Strukturen für Warn- und Sicherheitsweise findet. Hier muß die Basisstruktur modifiziert werden.

4 Fazit

Mit DITA hat das Thema XML in der technischen Dokumentation einen weiteren Schub erhalten. In Unternehmen wird stärker darüber nachgedacht, ob und wie XML dazu beitragen kann, die Prozesse zu optimieren und die Strukturen zu vereinheitlichen. Das Herzstück eines XML-basierten Dokumentationsprozesses bilden nicht die verwendeten Werkzeuge sondern die zugrundeliegende DTD (Schema). In vielen Industrien leben die Dokumente ungleich länger als die Werkzeuge mit denen sie initial erstellt wurden. Augenmerk ist daher stets auf die Struktur zu legen.

Es ist zu empfehlen, genau anzusehen, wo DITA paßt und wo Modifikationen notwendig sind und ob diese Modifikation in das DITA-Konzept passen. Wer konsequent den DITA-Weg gehen will, muß die Grenzen der Spezialisierung beachten. Es wäre allerdings ein großer Fehler, allein dem Hype von DITA zu folgen. Kompatibilität zu DITA ist einzig und allein dann erforderlich, wenn der Datenaustausch mit anderen Unternehmen eine wichtige Rolle spielt. Es ist auch nicht verwerflich, sich einzig und allein von DITA inspirieren zu lassen und völlig eigene Strukturen zu entwickeln.