



Erzeugung modularisierter und verlinkter Hypertextsichten in der *HyTex*-Pilotversion (Stylesheet `A11XML2HTML.xsl`)

Dokumentation

Projekt
**Hypertextualisierung auf
textgrammatischer Grundlage**
(www.hytext.info)

Benjamin Birkenhake
Eva Anna Lenz

2004

- 1 Überblick
- 2 Aufgabenkatalog
- 3 Umsetzung der Anforderungen
- 4 Verknüpfung zu anderen Dateien
- 5 Mögliche zukünftige Komponenten
- 6 Fazit

1. Überblick

Das Stylesheet `AllXML2HTML.xsl` ist das zentrale Stylesheet des HyTex-Projekts, das die beiden Teilaufgaben der Hypertextualisierung, nämlich einerseits die Segmentierung (Modularisierung) eines Textes, und andererseits dessen (Re-)Konnexion übernimmt. Als Eingabe bekommt das Stylesheet Regeln zur Modularisierung und die annotierte Textbasis, aus denen es modularisierte und HTML-basierte Hypertextsichten erzeugt.

Die Textbasis wurde vorher auf drei Ebenen annotiert: die logische Textstruktur (1), die Ebene der Definitionen und Termverwendungsinstanzen (2), und die Ebene der Koreferenz-Bezüge und Konnektive (3). Alle drei Ebenen wurden getrennt voneinander annotiert und validiert und anschließend mit Werkzeugen des Partner-Projekts A2 zusammengeführt (unifiziert), so dass alle drei Annotationsebenen schließlich in einer einzigen XML-Datei vorlagen, die das Stylesheet `AllXML2HTML.xsl` verarbeitet. Auch die Verarbeitung aller drei Ebenen kann im Prinzip getrennt durchgeführt werden: Für jede der drei Ebenen wurde ein eigenes Stylesheet entwickelt, welches die Details der Transformation der jeweiligen Ebene zu einer HTML-Präsentation steuert. Das Stylesheet `AllXML2HTML.xsl` bindet diese drei einzelnen Stylesheets ein und erzeugt anhand von Modularisierungsregeln Modul-Sichten. Die vorliegende Dokumentation beschreibt die dazu verwendeten Techniken. Die drei Annotationsebenen und ihre Präsentation werden dagegen in den folgenden Dokumentationen beschrieben:

- (1) logische Textstruktur: *Dokumentation zur Annotationsschicht: Dokumentenstruktur* (Eva Anna Lenz und Harald Längen)
- (2) Definitionen und Termverwendungsinstanzen: *Annotation definitorischer Textsegmente und »terminologiesensitives Linking«* (Michael Beißwenger) und *Annotationsschicht: Definitionen und Termverwendungsinstanzen* (Eva Anna Lenz und Michael Beißwenger)
- (3) Koreferenzen und Konnektive: *Strategien zur Herstellung kohäsiv autonomer Modulsichten* (Eva Anna Lenz und Angelika Storrer) und *Spezifikation für ein Annotationsschema für Koreferenzphänomene im Hinblick auf Hypertextualisierungsstrategien* (Anke Holler)

Die Modularisierung wird durch Modularisierungsregeln gesteuert, die in der Dokumentation *Segmentierungsregeln zur Erzeugung von Modul-Sichten* (Eva Anna Lenz) beschrieben sind.

Die Aufgabe des Stylesheets ist also viergeteilt. Die ersten beiden der vier Aufgaben sind die technisch bedeutsamsten. Sie regeln die Modularisierung und die typographische Präsentation der Ausgangsdaten. Die beiden folgenden Aufgaben sind für die Zielsetzung des Projektes am bedeutsamsten. Sie koordinieren die Herstellung der kohäsiven Geschlossenheit.

Die erste Aufgabe besteht darin, die Ausgangsdaten gemäß extern repräsentierter Regeln in Module zu zerlegen. Da bei Hypertextualisierungen von Texten, die nicht als Hypertexte konzipiert wurden, die Frage, nach welchen Prinzipien die Ausgangsdaten modularisiert werden sollten von zentraler Bedeutung ist, bietet die Wahl einer externen Steuerung über eine definierte Schnittstelle die Möglichkeit, die Modularisierung schnell und einfach zu beeinflussen.

Die zweite Aufgabe ist ebenso grundlegend wie notwendig. Zur erfolgreichen Präsentation von textuellen Daten ist es unerlässlich, grundlegende logische Textstrukturen wie Überschriften, Absätze u.ä. typographisch entsprechend der gängigen Konventionen zu präsentieren. Im Aufbau des Projektes wird daher eine Informationsebene in den Ausgangsdaten benötigt, die diese Strukturen abbildet, sowie eine Transformation in ein adäquates Präsentationsformat. Als Ausgangsdatenformat wird in diesem Falle ein modifiziertes DocBook-Format verwendet und als Präsentationsformat HTML. Dieses Stylesheet regelt u.a. die Transformation vom ersteren ins letztere.

Die dritte Aufgabe besteht in der Auswertung der in den Ausgangsdaten enthaltenen expliziten Definitionen und Termverwendungen. Diese Informationsebene wurde gemäß einer für dieses Projekt und diese Anforderung erarbeiteten Dokumententypdefinition erstellt. Termverwendungen sollen dabei sowohl auf Definitionen des Terms im Text als auch auf den Eintrag des Terms im Termnet verweisen. Diese Verweise werden über Popups und Links in HTML umgesetzt.

Die vierte Aufgabe besteht in der Auswertung der in den Ausgangsdaten explizit annotierten Koreferenz-Bezüge und Konnektoren. Diese Informationsebene wurde ebenfalls gemäß einer für dieses Projekt und diese Anforderung erstellten Dokumententypdefinition erstellt. Koreferenzen zwischen unterschiedlichen Bestandteilen des Textes werden mit unterschiedlichen, an die Art der Koreferenz angepassten hypertextuellen Mechanismen präsentiert. Unter diese Mechanismen fallen Links, Popups, Tooltips und Sichtfelderweiterungen, d.h. Einblendungen zuvor unsichtbarer Module.

Um den komplexen Anforderungen der Vorgaben gerecht zu werden, sieht das Stylesheet mehrere Transformationsschritte vor, die nacheinander abgearbeitet werden.

2. Aufgabenkatalog

1. Die HTML-Präsentation jedes Dokuments soll in nur einer HTML-Datei gespeichert werden. (*Prinzip*: Generierung modularisierter Hypertext-Sichten unter Erhaltung des Ausgangsdokuments *anstatt* Zerlegung in des Ausgangsdokuments in Einzeldokumente und Rekonexion durch Links).
2. Die Präsentation der unterschiedlichen Module eines Dokuments soll dabei über HTML, CSS und Javascript gelöst werden.
3. Die Generierung modularisierter Sichten auf die Inhalte der einzelnen Dokumente soll dabei durch die Datei `rules.xml` gesteuert werden.
4. Die Transformation zur HTML-Präsentation soll sowohl mit den nicht unifizierten, einzelnen XML-Dokumenten als auch mit den unifizierten Daten funktionieren. Für jede Annotationsebene sollen separate Stylesheets entwickelt werden, die in ein Gesamt-Stylesheet eingebunden werden. Die Einzel-Stylesheets sollen nach Möglichkeit so aufgebaut sein, dass in Ihnen eine »normale« Transformation der XML-Ausgangsdaten nach HTML enthalten ist, wie sie gestaltet wäre, wenn man lediglich eine der Ebenen in ein statisches HTML-Dokument transformieren wollte. Auf diese Weise können die Ebenen getrennt voneinander annotiert werden, die einzelnen Stylesheets bleiben übersichtlicher und sowohl die Stylesheets als auch die Ebenen können getrennt voneinander getestet werden.

Die Modularisierung über Javascript [s. Anforderung 1.] sowie die Links vom Termnet in die Module sollen von diesen Stylesheets getrennt werden.

3. Umsetzung der Anforderungen

Anforderung 1: vollständig umgesetzt

Jedes XML Dokument wird in eine HTML-Datei transformiert.

Der eigentliche Inhalt des Dokuments, d.h. der Text, ist tatsächlich physisch in einer HTML-Datei.

Ausgliedert aus dieser Datei wurde lediglich das CSS-Stylesheet, das einen Teil der Präsentation kontrolliert.

Anforderung 2: vollständig umgesetzt

Da die Vorgaben für die Präsentation verlangten, zwar das gesamte Dokument in einer HTML-

Datei zu speichern, bei der Präsentation aber jeweils nur einen Teil des gesamten Dokuments (nämlich einzelne Hypertext-Module) anzuzeigen (Prinzip der Modul-Sichten, s.o.), war es erforderlich, nicht-aktive Module mittels CSS auszublenden und das gesamte Linking und die Navigation, d.h. Menüs, Inhaltsverzeichnisse, Vorwärts- und Rückwärts-Links sowie alle vom Autor gesetzten und alle automatisch erzeugten, kohäsionsunterstützenden Links über Javascript zu steuern. Diese Aufgabe wurde technisch wie folgt gelöst:

- Jedes Modul innerhalb des Dokuments erhält einen für dieses Dokument eindeutigen Bezeichner, in diesem Fall ein `id`-Attribut.
- Alle Module bekommen zudem das Attribut `style`, das als Wert `visibility:hidden; display:none;` hat, was zur Folge hat, dass beim Aufruf der Datei zunächst alle Module ausgeblendet werden. Die einzige Ausnahme von dieser Regel bildet das zweite `div`-Element, welches das Inhaltsverzeichnis enthält, das dem Benutzer beim ersten Aufruf der HTML-Datei als Übersichtshilfe angezeigt werden soll.
- Alle Links, die die Bewegung von Modul zu Modul innerhalb dieses Dokuments erlauben, verweisen nicht auf eine URL, sondern führen bei Betätigung zum Aufruf einer Javascript-Funktion mit dem Wert des `id`-Attributs desjenigen Moduls, das angezeigt werden soll. Die entsprechende Javascript Funktion blendet dann zunächst alle Module aus, und dann dasjenige Modul, dessen `id` übergeben wurde, wieder ein.

Anforderung 3: vollständig umgesetzt

Um die komplexen Anforderungen an die gewünschte Transformation erfüllen zu können, ist das Stylesheet über »Modes« in drei Transformationsschritte geteilt. Der erste dieser Schritte bindet die externe Datei `rules.xml` ein und erzeugt auf Basis der darin definierten Regeln die Grenzen zwischen den einzelnen Modulen. Die Regeln, die in der Datei `rules.xml` zu finden sind, können sich dabei theoretisch auf beliebige Annotationsebenen beziehen, so dass eine Modularisierung nach unterschiedlichen Kriterien möglich ist. Im HyTex-Projekt wird jedoch nur die Ebene der logischen Textstruktur (DocBook) zur Modularisierung herangezogen. Das bedeutet, dass das Skript in den unifizierten Daten wenigstens die Doc-Book-Ebene benötigt, um eine erfolgreiche Modularisierung durchzuführen. Wie in den Anforderungen beschrieben, können beliebige weitere Annotationsebenen hinzugefügt und Modularisierungsregeln dafür aufgestellt und umgesetzt werden.

Anforderung 4: weitestgehend umgesetzt

Die Transformationsregeln der einzelnen Ebenen werden jeweils in die Stylesheets `Docstruc2HTML.xsl`, `Def-Term2HTML.xsl` und `Koref2HTML.xsl` ausgelagert. Auch können in diesen Dateien praktisch »normale« Transformationsregeln angegeben werden. Einzige Einschränkung ist, dass alle Templates (welche die Transformationsregeln definieren) im Mode `divs2html` oder einer Erweiterung dieses Modes in der Form `divs2html-Erweiterung` stehen. Entsprechend müssen auch alle `apply-templates` in den entsprechenden Modi stehen.

Da – wie im vorhergehenden Abschnitt bereits erwähnt – die Transformation der Module nach HTML nicht gänzlich ebenenunabhängig läuft, kann diese Anforderung nur in Teilen realisiert werden: Zur Transformation wird immer die Ebene der logischen Dokumentstruktur benötigt, da an diese die höheren HTML-Elemente, wie `html`, `head` (sowie dessen Kinder) und `body` geknüpft sind. Die Ebene der logischen Dokumentstruktur lässt sich allerdings sehr wohl allein transformieren und in Kombination mit ihr lassen sich auch alle weiteren (teilunifizierten) Daten transformieren. Diese Anforderung lässt sich aber evtl. mit weiteren Arbeiten vollständig umsetzen.

4. Verknüpfungen zu anderen Dateien

Das Stylesheet `AllXML2HTML.xsl` ist explizit mit folgenden anderen Komponenten verbunden:

<code>XML-Korpus:</code>	Dieses Stylesheet arbeitet auf den XML-Korpus-Dateien.
<code>rules.xml:</code>	Dieses Stylesheet bettet die Datei <code>rules-generated.xml</code> ein, um damit die Modulgrenzen zu bestimmen.
<code>Docstruc2TOC.xsl:</code>	Dieses Stylesheet bettet die Datei <code>Docstruc2TOC.xsl</code> ein, um auf Grundlage der <i>DocBook</i> -Struktur automatisch ein Inhaltsverzeichnis zu erstellen.
<code>Docstruc2HTML.xsl:</code>	Diese Stylesheet bettet die Datei <code>Docstruc2HTML</code> ein, welche die Transformationsregeln von der <i>DocBook</i> -Ebene nach HTML enthält.
<code>Def+Term2HTML.xsl:</code>	Dieses Stylesheet bettet die Datei <code>Def+Term2HTML.xsl</code> ein, welche die Transformation von der Term- und Definitionsebene nach HTML enthält. Hierin wird auch der Popup-Mechanismus definiert.
<code>baseName2fileName.xsl:</code>	Dieses Stylesheet bettet die Datei <code>baseName2fileName.xsl</code> ein. Hierin wird aus einem Term ein einheitlicher Dateiname generiert, was die Schnittstelle zum Glossar darstellt, welches aus <i>TermNet</i> erzeugt wird.
<code>Koref2HTML.xsl:</code>	Dieses Stylesheet bettet die Datei <code>Koref2HTML.xsl</code> ein, die die Transformation von der Koreferenz- und Konnektionsebene nach HTML definiert. Hierin wird der Sichtfelderweiterungs-Mechanismus definiert.

5. Mögliche zukünftige Komponenten

Alternative Präsentation:

Da das Paradigma der »einen HTML-Datei« einige Nachteile mit sich bringt (z.B. große Dateien und Javascript-Abhängigkeit), böte sich bei einer Erweiterung dieses Stylesheets v.a. eine Präsentation an, die serverseitig die einzelnen Module dynamisch zusammenfügt. Hierfür wäre z.B. ein PHP- oder Perl-basiertes System denkbar.

Alternativ dazu wäre auch eine statische, aus vielen HTML-Dateien bestehende Variante denkbar.

Ergänzende Ebenen:

Ergänzend zu den bestehenden Ebenen »DocBook«, »Definitionen und Termverwendungsinstanzen« und »Koreferenz und Konnektive« könnten theoretisch weitere Ebenen eingebunden werden. Ihre Bearbeitung kann jeweils in ein Einzel-Stylesheet ausgelagert werden. Voraussetzung ist jedoch die Unifizierbarkeit der jeweils kombinierten Ebenen.

Alternative Ausgangsdaten:

Da die einzig für die Generierung der Modulsichten verwendete Ebene die »Docbook«-Ebene ist, die »nur« die logische Textstruktur repräsentiert, welche in praktisch allen Formen von Texten zu finden ist, lassen sich auch Texte, die nicht im Projektkorpus enthalten sind, prinzipiell mit dem

Stylesheet transformieren (sofern ihre Textstruktur in entsprechendes Markup überführt wird). Würde das Projektkorpus erweitert, könnten somit auch neu hinzukommende Dokumente problemlos anhand des Stylesheets für die Präsentation transformiert werden.

6. Fazit

Das Stylesheet `AllXML2HTML.xsl` ist neben dem Stylesheet `XTM2SVG.xsl`¹ das zentrale Stylesheet für die Präsentation der Daten. Es ist so aufgebaut, dass es nicht nur leicht erweiterbar, sondern zudem auch für andere Projekte nutzbar ist.

1 Vergleiche hierzu die Dokumentation: *Generierung von Glossarsichten mit dem Stylesheet XTM2SVG.xsl* (Benjamin Birkenhake; <http://www.hrz.uni-dortmund.de/~hytex/hytex/Publikationen/generierung-glossarsichten.pdf>).