Aufgabe 7: Kodierung von Varianz mit Parallel Segmentation (Vielschwetzen)

Christof Schöch

DARIAH-DE
Lizenz: Creative Commons Attribution 4.0 (CC-BY).
Das vorliegende Dokument wurde für DARIAH-DE neu erstellt. Das vorliegende Dokument beruht teilweise auf den folgenden Materialien: James Cummings, Renée Baalen, Ylva Berglund-Prytz: "An Introduction to XML and the Text Encoding Initiative", Digital.Humanities@Oxford Summer School 2012, University of Oxford, 2012, http://digital.humanities.ox.ac.uk/dhoxss/2012/, CC-BY-SA.

Transkribieren von Primärquellen mit TEI

Einleitung

1. Lernziele

* Praktische Erfahrung mit der Anwendung von Elementen wie `app`, `rdg` und `lem`
* Verständnis, wie Informationen im `teiHeader` und im `body` zusammenhängen
* Weitere Vertrautheit mit Oxygen

2. Materialien

* PDF-Datei (`Vielschwetzen-Faksimiles.pdf`): enthält die digitalen Faksimiles der zweiten Seite des Kapitels "Von vielschwetzen" aus dem Narrenschiff von Sebastian Brant, und zwar in sechs verschiedenen Fassungen (1494 Münchner Exemplar, 1494 Berliner Exemplar; 1506; 1512; 1560; 1872 – in dieser Reihenfolge in der Datei).
* TXT-Datei (`Vielschwetzen-Ausgaben.txt`) mit minimalen Metadaten zu den verschiedenen Textfassungen.
* TXT-Datei (`Vielschwetzen-Transkription.txt`) mit den Transkriptionen der Zeilen 5-8 aus den sechs verschiedenen Textfassungen.
* XML/TEI-Datei (`Vielschwetzen-Start.xml`) als Ausgangsbasis für die Kodierung der verschiedenen Textfassungen.

3. Überblick

In dieser Übung analysieren wir zunächst die Unterschiede, die sich zwischen den sechs Textfassungen feststellen lassen. Wir dokumentieren dann die sechs Textfassungen, die der Kodierung zugrundeliegen, im `teiHeader`. Dann kodieren wir die Texte und ihre Varianz in TEI mit der "parallel segmentation"-Methode.

Erster Teil: Vorbereitung des `teiHeader`

1. Materialien sichten

Öffnen Sie den Ordner mit den Materialien für diese Übung (ggfs. zunächst das ZIP-Archiv entpacken). Schauen Sie sich zunächst einmal das digitale Faksimile (`Narrenschiff-Vielschwetzen.pdf`) und die Datei mit den Metadaten (`Vielschwetzen-Ausgaben.txt`) an.

* Identifizieren Sie die für unsere sechs Fassungen relevanten Katalogeinträge
* Mit Bezug auf die Metadaten-Datei: Welche Informationen sollte man im Header des TEI-Dokuments mindestens festhalten, damit alle Ausgaben und Fassungen klar identifiziert sind?

2. Im `titleStmt` den Herausgeber eintragen

Öffnen Sie nun die Datei "Vielschwetzen-Start.xml" mit Oxygen. Sie werden sehen, dass diese Datei bereits einige Dinge enthält; insbesondere kann man sehen, dass es einen `teiHeader` gibt (oberer Teil) und einen `text`-Bereich (unterer Teil). Wir sollten nun als erstes den `teiHeader` um einige Informationen ergänzen.

* Finden Sie Header das Element `titleStmt`: Sie sehen, dass dort schon ein paar Informationen eingetragen sind.
* Finden Sie das Element `editor` und tragen Sie dort Ihren Namen ein; Sie bearbeiten ja nun dieses Dokument. Überprüfen Sie, ob das Dokument weiterhin valide ist! (`Strg+Shift+V`).
* Speichern Sie die Datei unter einem neuen Namen ab.

3. Liste der Textzeugen anlegen

* Finden Sie nun das Element `sourceDesc`, das zur Beschreibung der Quellen einer Edition oder Transkription dient. Sie sehen dort einen Platzhalter; löschen Sie den Text `[Beschreibung der Quellen]`, inklusive des öffnenden und schließenden `p`-Tags.
* Überprüfen Sie erneut, ob das Dokument weiterhin valide ist: `Strg+Shift+V`; Ergebnis? Versuchen Sie, die Fehlermeldung ein Stück weit zu verstehen.
* Fügen Sie in die Lücke zwischen dem öffnenden und dem schließenden `sourceDesc`-Tags folgende Zeichen ein: "<listW" Was passiert? Wenn Sie die Auswahl `listWit` vorgeschlagen bekommen, bestätigen Sie mit der Enter-Taste.
* Gehen Sie nun mit dem Cursor zwischen die beiden `listWit`-Tags und fügen sie mit der Enter-Taste eine neue Zeile ein. Tippen Sie dann: "<" und wählen Sie aus den vorgeschlagenen Elementen "witness" aus. Bestätigen Sie mit Enter.
* Fügen Sie nun als Inhalt des `witness`-Elements eine kurze Beschreibung des ersten Textzeugen ein; bespielsweise: "Narrenschiff, Ausgabe Basel 1494,

Münchner Exemplar."

* Ist Ihr Dokument jetzt wieder valide? Überprüfen Sie das.

4. Weitere Textzeugen eintragen

Bisher haben Sie nur einen ersten Textzeugen eingetragen.

* Wiederholen Sie nun die letzten beiden Schritte (`witness`-Element und Inhalt eintragen) für die weiteren fünf Textzeugen: das Berliner Exemplar der Ausgabe von 1494 sowie die Ausgaben von 1506, 1512, 1560 und 1872.

5. Die Textzeugen eindeutig identifizieren

Jetzt haben wir zwar eine Liste der Textzeugen, aber noch keine maschinen-lesbare Identifikation. Dafür müssen wir jedem `witness`-Element eine `xml:id` geben, also ein einzigartiges Label. Dies geschieht mit der Attribut-Funktion von XML/TEI.

* Gehen Sie zum ersten Eintrag in der `witness`-Liste. Gehen Sie mit dem Cursor ans Ende des öffnenden `witness`-Tags, aber noch vor die schließende spitze Klammer; hier: <witness|>.

* Fügen Sie ein Leerzeichen ein; fügen Sie dann ein "x" ein; in der Auswahlliste wählen Sie nun `xml:id` und bestätigen mit der Enter-Taste. Was passiert?

* Geben Sie nun dem ersten Textzeugen ein Label; mein Vorschlag wäre, dass Sie das Label folgendermaßen aufbauen: erst "Ed" für Edition, dann das Jahr, und wenn notwendig einen weiteren Buchstaben zur Identifikation des Exemplars. Für das Münchner Exemplar der Ausgabe von 1494 wäre das also folgendes Label: "Ed1494M".

* Geben Sie nun den anderen Textzeugen ebenfalls ein entsprechendes Label; behalten Sie das System der Labels bei und achten Sie darauf, dass jeder Textzeuge ein einzigartiges Label bekommt.

* Ist Ihr Dokument weiterhin valide? Wenn ja, sehr gut!

Damit haben wir die Grundlagen gelegt, um später in der Transkription einzelne Lesarten einem Textzeugen eindeutig zuzuordnen.

6. Die Kodierungs-Methode angeben

Bevor wir mit der Kodierung der verschiedenen Textzeugen anfangen können, müssen wir noch die Methode angeben (Sie erinnern sich an das Input-Referat). Dies geschieht im Bereich des Headers, der `encodingDesc` heißt.

* Löschen Sie den Platzhalter einschließlich der `p`s.

* Fügen Sie nun in der verbleibenden Leerzeile ein ">" ein und warten Sie auf die Vorschläge von Oxygen. Wählen Sie aus der Liste das Element "variantEncoding" aus und bestätigen Sie mit der Enter-Taste. Oxygen ergänzt das Element und zwei Attribute (diese sind verpflichtend).

* Fügen Sie im ersten Attribut, method, zwischen den Anführungszeichen "parallel-segmentation" ein.

* Fügen Sie im zweiten Attribut, location, zwischen den Anführungszeichen

"internal" ein.

* Überprüfen Sie, ob Ihr Dokument weiterhin valide ist. Wenn nicht, lesen Sie die Fehlermeldung um herauszufinden, was nicht stimmt.

Jetzt sind wir soweit, dass wir mit der eigentlichen Kodierung der Varianten beginnen können.

Zweiter Teil: Kodierung der varianten Textfassungen

1. Vorbereitung

Die PDF-Datei "Vielschwetzen.pdf" enthält digitale Faksimiles von verschiedenen Ausgaben des Narrenschiffs. Vergleichen Sie einmal den Text und das Layout dieser Ausgaben. Was fällt Ihnen auf? Öffnen Sie nun außerdem die Datei "`Vielschwetzen-Transkription.txt`" mit Oxygen. Diese Datei enthält die Transkription der vier Zeilen, die wir nun kodieren wollen, und zwar für jeden der fünf Textzeugen.

* Schauen Sie sich auch hier genau die Unterschiede an.

* Die Transkription ist am Ende dieses Aufgabenblattes auch angehängt. Sie können dort von Hand die Stellen markieren, die sich in den Textfassungen voneinander unterscheiden.

2. Kodierung des Basistextes der ersten Zeile

Wir kodieren den Text nun Zeile für Zeile. Die Struktur einer solchen Kodierung sieht wie folgt aus: <l> <app> <lem></lem> <rdg></rdg> <rdg></rdg> </app> </l> Ein paar Hinweise dazu:

* Die oberste Ebene ist die Zeile selbst, die man mit `l` kodiert. (Text des Basistexts am Versanfang oder Versende, für den keine Varianz besteht, kann dort direkt markiert werden.)

* Die nächste Ebene eröffnet den Apparat-Eintrag, der hier im laufenden Text erfolgt.

* Die nächste Ebene enthält einerseits den Basistext oder präferierten Text (`lem` = "lemma"), andererseits die Varianten (`rdg` = "readings").

* Dann werden die Hierarchiestufen wieder geschlossen.

Diese Struktur müssen wir nun im TEI-Dokument erstellen und mit Inhalt füllen. Dabei gehen wir davon aus, dass wir einer traditionellen editorischen Schule angehören und die erste Ausgabe, hier also die Baseler Ausgabe von 1494 (Münchner Exemplar) als Basistext annehmen.

* Erstellen Sie also zunächst nach dem Kommentar " <!-- (Hier kodieren wir die Zeilen 5-8) -->" im TEI-Dokument eine neue Verszeile: <l></l>

* Geben Sie der Zeile außerdem eine Nummer; fügen Sie als Attribut "n" ein, und als Wert des Attributs "5", denn wir beginnen ja mit der Zeile 5.

* In der ersten Zeile, die wir kodieren wollen, besteht für die ersten beiden Worte keinerlei Varianz zwischen den Textzeugen. Fügen Sie also nach dem öffnenden `l`-Tag den Text "Wer reden " ein.

* Danach geht es los mit den Varianten. Fügen Sie also nun eine neue Zeile und dort das Element `app` ein.

* Zwischen dem öffnenden und dem schließenden `app`-Tag fügen Sie das Element `lem` ein.

* Als Inhalt von `lem` geben Sie den Rest der Zeile, wie er in der Ausgabe von 1494, Münchener Exemplar, zu finden ist ein; also: "wil/so er nit sol". Danach sollte das schließende </lem>-Tag stehen.

Wir müssen nun auch angeben, woher diese Lesart stammt. Das geschieht mit einem Attribut zu dem Element `lem`.

* Gehen Sie mit dem Cursor an das Ende des öffnenden `lem`-Tags, aber noch vor die schließende Spitze Klammer; hier: <lem|>.

* Fügen Sie dort ein Leerzeichen ein, und geben Sie das Attribut "`wit`" ein. Wenn Sie nur "w" tippen und dann "wit" aus der Vorschlagsiste wählen und mit Enter bestätigen, wird gleich noch ="" eingefügt.

* Bleiben Sie mit dem Cursor zwischen die beiden Anführungszeichen; es werden Ihnen eine Reihe von Werten vorgeschlagen; das sind die xml:ids, die wir im Header definiert haben.

* Wählen Sie die passende xml:id aus (hier also "Ed1494M") und bestätigen Sie mit der Enter-Taste.

3. Kodierung weiterer Lesarten der ersten Zeile

Prüfen Sie, ob noch andere Textfassungen die gleiche Lesart belegen. Wenn ja, sollten Sie die entsprechenden xml:ids ebenfalls noch als weitere Werte des Attributs "wit" angeben:

* Fügen Sie einfach vor dem schließenden Anführungszeichen ein Leerzeichen ein und geben Sie dort eine weitere xml:id an; tun Sie dies für "Ed1494B" und "Ed1506".

Für die weiteren Lesarten fügen wir nun in einer jeweils neuen Zeile `rdg`-Elemente ein.

* Fügen Sie nach dem schließenden </lem>-Element eine neue Zeile und ein weiteres `rdg`-Element ein.

* Die Ausgabe von 1560 hat eine minimal abweichende Lesart; tragen Sie als Inhalt des `rdg`-Elements die Transkription der Zeile im nächsten Textzeugen ein, wobei Sie natürlich "Wer reden" weglassen, da dies ja schon als invarianter Teil der Zeile kodiert ist.

* Geben Sie auch diesem `rdg`-Element ein Attribut "wit", in dem Sie die xml:id der Ausgabe von 1560 als Wert angeben.

* Wiederholen Sie den gesamten Vorgang (neue Zeile, neues `rdg`-Element, Transkription und wit-Attribut mit xml:id als Wert) in analoger Weise auch für die bisher noch nicht berücksichtigten.

* Überprüfen Sie, ob das Dokument weiterhin valide ist.

Sie haben die erste Zeile in ihren drei Varianten aus sechs Textzeugen kodiert! Sehr gut!

4. Kodierung weiterer Zeilen

Wiederholen Sie den gesamten Vorgang (Schritte 2 und 3) nun auch für die nächste Zeile. Wenn Sie damit besonders schnell vorankommen, können Sie es auch für die Zeilen 6 und 7 machen. Achtung: In Zeile 2 gibt es einen besonderen Buchstaben, der sich nicht einfach so Transkribieren lässt. Genau genommen handelt es sich um ein kleines u mit kleinem e darauf; wir kodieren dieses Zeichen vorerst annäherungsweise mit der Entity ů die dem Zeichen aber nicht genau entspricht. Genauere Informationen finden Sie auf den Seiten des UNICODE Konsortiums, unter www.unicode.org/charts, insbesondere im Bereich Latin Extended A. Überlegen Sie, ob es nicht auch noch eine einfachere, wenn auch deutlich weniger präzise Lösung geben könnte.

5. Schon fertig? Dann gibt es hier noch eine Extra-Aufgabe

Wenn Sie nicht die Transkriptionen, sondern die Faksimiles anschauen, könnten Ihnen in den vier Zeilen noch weitere Unterschiede im Detail der Schreibung der Buchstaben auffallen. Finden Sie auf der Seite von UNICODE heraus, ob und wie man diese Zeichen korrekt kodieren könnte. Ende der Aufgabe!

Anhang: Transkription der vier Zeilen aus "Von Vielschwetzen"

Seite 1 des PDFs: Ausgabe von 1494, Berliner Exemplar [Erste Seite; nur zur Information; lassen wir aus] Seite 2 des PDFs: Ausgabe von 1494, Berliner Exemplar Wer reden wil/so er nit sol Der fügt in narren orden wol Wer anttwurt/ee man froget in Der zeigt sich selbs eyn narren syn Seite 3: Ausgabe von 1494, Münchner Exemplar Wer reden wil/so er nit sol Der f[ue]gt in narren orden wol [ue] => wir kodieren annäherungsweise mit Entity ů Wer antwürt/ee man froget in Der zeigt sich selbs eyn narren syn Seite 4: Ausgabe von 1506 Wer reden wil/so er nit sol Der f[ue]gt in narren orden wol (siehe oben) Wer antwurt/ee man frogt in Der zeigt sich selbs eyn narren syn Seite 5: Ausgabe von 1512 Wer reden/wil so er nit sol Der f[ue]gt in narren orden wol (siehe oben) Wer antwurt/ee man fragt in/ Der zeigt sich selbs ein narren syn Seite 6: Ausgabe von 1560 Wer reden wil/so er nicht sol/ Der fügt in Narren orden wol. Wer antwort eh man fraget in/ Der zeigt sich selbst ein Narren sin. Seite 7: Ausgabe von 1872 Wer reden will wo er nicht soll, Der taugt zum Narrenorden wohl, Denn wer antwortet ungefragt, Zeigt wie ihm Schellenklang behagt.