XML-TEI-kodierte Texte editieren

Literarische Texte, die mit elektronischen Mitteln formal nachbearbeitet werden müssen, sind idealerweise XML-kodiert gespeichert. Passende XML-Schemata dafür liefert die Text Encoding Initiative (TEI). Linux zeigt dabei im Editionsprozess seine Stärken als Betriebssystem für die einzelnen Arbeitsplätze der Bearbeiter. Harald Jele

Die Text Encoding Initiative (TEI) ist ein Konsortium, das sich seit 1994 um den Entwurf, die Entwicklung und die Verbreitung von Standards kümmert, die zum elektronischen Speichern und weiteren Verarbeiten von Texten herangezogen werden. Diese Standards sind offen zugänglich. [1]
Die Mitglieder des Konsortiums stammen vorwiegend aus dem universitären Umfeld sowie aus Forschungseinrichtungen, die sich mit den Themen rund um die maschinelle Verarbeitung von Texten befassen. Dabei liegen die Schwerpunkte auf Texten der Geisteswissenschaften, der Sozial- und Wirtschaftswissenschaften und der Linguistik.
Die Erkenntnisse, die sich aus dem Einsatz dieser Standards ergeben, fließen kontinuierlich in ihre Weiterentwicklung ein.
Obwohl dieses Wissen seit rund 20 Jahren vorhanden und in den einschlägigen Kreisen bekannt ist, wird es im Grunde immer noch recht häufig ignoriert. Dass damit ganze Editionsprojekte in eine Sackgasse mit absehbaren Folgen manövriert werden, wird vielfach in Kauf genommen. Viele Gründe, warum dies so ist sind darauf zurückzuführen, dass Bearbeiter den Umgang mit dem XML-kodierten Text scheuen und lieber auf eine Oberfläche zurückgreifen, die sie aus den gängigen Textverarbeitungen kennen – ungedenk dessen, wie die Texte dabei kodiert werden.
Moderne XML-basierte Arbeitsumgebungen helfen hier aus und bieten unterschiedlichste Sichtweisen auf den zu editierenden Text sowie eine Vielzahl an Werkzeugen, die einerseits das Bearbeiten erleichtern und andererseits helfen sollen, die Kodierung des Quelltexts nicht aus den Augen zu verlieren.
Dass Texte, die mit einer Textverarbeitung erstellt wurden, mit wenig Aufwand in ein XML-TEI-kodiertes Dokument überführt werden können, zeigen die Konvertierungsroutinen der „OxGarage“ [2].

Linux, der XML-Primus?

Die Kodierung, Speicherung und Weiterverarbeitung unterschiedlichster Informationen in Form von XML-kodierten Dateien ist im Betriebssystem des Pinguins eine bereits lang gelebte und auch tief verwurzelte Tradition.
Dieser Umstand lässt einen vermuten, dass auch entsprechende Werkzeuge vorhanden und offen zugänglich sind, mit denen solcherlei Daten manuell und/oder maschinell nachbearbeitet werden können. Ist diese Vermutung zutreffend, so sollte

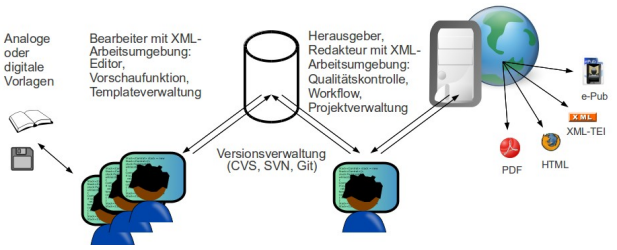

Abbildung 1: Typische Vorgänge im Editionsprozess eines (literarischen) Werkes.

es einem möglich sein, sich baukastenartig alle benötigten Werkzeuge aus den gängigen Repositorien der Linux-Distributoren zu einer tauglichen Arbeitsumgebung selbst zusammenzustellen. Diese Vorgehensweise bringt den Vorteil mit sich, dass man auf bereits Vertrautes zurückgreifen kann.

In Abbildung 1 sind die typischen Zusammenhänge im Editionsvorgang dargestellt:
Ein Bearbeiter redigiert einen Text aufgrund einer analogen oder digitalen Vorlage. Dieser wird zumeist auf der Grundlage von XML-Templates bearbeitet und arbeitsteilig fertig gestellt. Das zentrale Werkzeug ist somit ein XML-Editor, der Templates verwalten und einbinden können sollte. Zudem ist eine Vorschaufunktion an dieser Stelle hilfreich. Mittels XSLT-Stylesheets, die von der TEI zur Verfügung gestellt werden [3], können Texte sofort in ihrer späteren Repräsentationsform (HTML, PDF oder e-Pub) betrachtet werden. Als XSLT-Processor kommt dazu in nahezu allen Fällen die Open-Source-Software Saxon [4] zum Einsatz.
Da die bearbeiteten Texte zumeist von mehreren Bearbeitern redigiert werden, müssen diese so abgelegt werden, dass eine sich ergänzende Zusammenarbeit möglich ist. Daher werden diese nicht in einem üblichen Dateisystem, sondern vielmehr innerhalb eines Systems zur Versionierung (CVS, SVN oder Git) gespeichert.
Ein Redakteur oder Herausgeber entnimmt der Versionsverwaltung den letzten Stand und gibt diesen nach diversen Arbeiten, die der Qualitätssicherung dienen, über einen Publikationsserver frei. Editionsprojekte greifen dabei häufig auf die OpenSource-Software Sade [5] (= Scalable Architecture for Digital Editions) der Berlin-Brandenburgischen Akademie der Wissenschaften oder die Software XTF [6] (= eXtensible Text Framework) der California Digital Library, die ebenso als Open-Source veröffentlicht wurde, zurück. Die Aufgabe

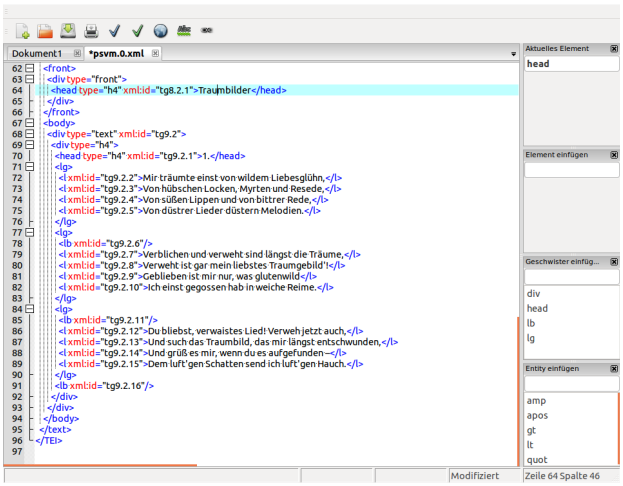

Abbildung 2: Eine XML-TEI-kodierte Datei im XML Copy Editor.

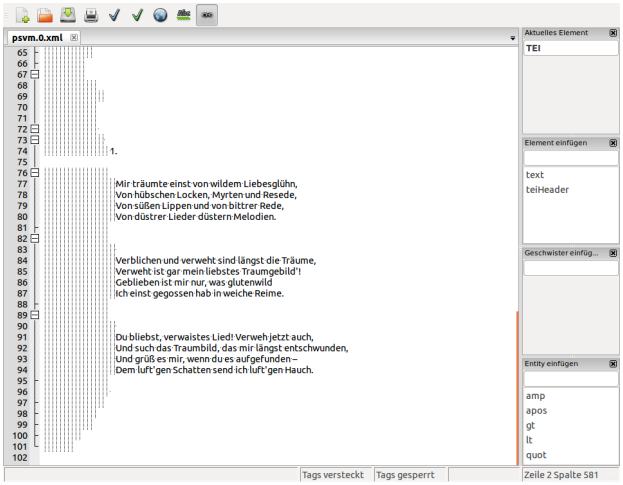

Abbildung 3: Die Anzeige einer Datei mit ausgeblendeten und gesperrten XML-Tags.

eines solche Publikationsservers ist die Repräsentation der Inhalte. Dies geschieht entweder innerhalb des Browsers (als HTML-Text) oder über spezielle Reader (PDF, XML-TEI und e-Pub).
Damit spielt der XML-Editor, der das Kernstück einer solchen Arbeitsumgebung bildet, eine zentrale Rolle. Bietet ein solcher keine Template-Verwaltung und keine Integration einer Vorschaufunktion, so muss beides außerhalb des Editors nachgebildet werden: Templates werden in Dateiverzeichnissen gespeichert und Saxon wird manuell gestartet.
Von der Betrachtung völlig ausgenommen wurden die „Generalisten“. Viele Editoren bieten Plugins an, mit denen das Bearbeiten von XML-Dateien durch ergänzende Funktionen unterstützt wird. Jemand, der über Jahre seinen Editor lieb gewonnen hat, wird sich daher wohl eher um ein passendes Plugin als um einen alternativen Editor umsehen. Eine Übersicht dazu findet sich unter [7].

XML-Editoren der Distributionen

Bei der Wahl eines geeigneten XML-Editors ergeben sich einige Überraschungen:
Die Recherche ergibt, dass laut Aussagen der Community der XML Copy Editor [8] neben Conglomerate [9] ein beliebter Editor ist. Ersterer zeigt einen deutlich größeren Funktionsumfang und ist bei der Verarbeitung großer Dateien wesentlich stabiler. Daher wird dieser im Weiteren betrachtet. Conglomerate stolpert häufig über seine Eigenschaft, ständig den zu bearbeitenden Text validieren zu wollen und stürzt daher gerne unvermittelt ab. Dateien, die keinen wohlgeformten XML-Code beinhalten, öffnet Conglomerate erst gar nicht. Damit ist er in vielen Fällen ungeeignet, denn im Vorgang des Editierens sind solche Zustände eigentlich durchaus die Regel.

Der XML Copy Editor lässt sich in Ubuntu mit

```
apt-get install xmlcopyeditor
```

einfach installieren – jedoch leider anschließend nicht starten. Die Installationspakete sind seit einigen Ubuntu-Versionen nicht mehr brauchbar. Eine selbst kompilierte Version sollte jedoch funktionieren.
Eine Anleitung dazu findet sich unter [10]. Daneben ist über die Homepage des Projekts ein Paket für die Installation auf einem 32bit-Ubuntu-System verfügbar, das zuverlässig funktioniert.

```
wget https://sourceforge.net/projects/xml-copy-editor/files/xml-copyeditor-linux/1.2.0.9/xmlcopyeditor_1.2.0.9-1_i386.deb
dpkg -i xmlcopyeditor_1.2.0.9-1_i386.deb
apt-get install -f
```

Die letzte Zeile installiert all jene Pakete nach, von denen der XML Copy Editor abhängt und die auf dem System nicht installiert waren.

Abbildung 2 zeigt das angenehm zurückhaltende Design des Editors. Nur die wichtigsten Funktionen sind direkt zugänglich, alles Weitere ist im Menü untergebracht und zudem über Funktionstasten erreichbar.
Die Struktur der Datei wird im Hauptfenster automatisch in einer schmalen Baumansicht dargestellt, ohne dass dadurch die Inhalte schlecht lesbar würden. Zusammengehörige Teile eines solchen Baums können einfach ein- oder ausgeklappt werden. Längere Zeileninhalte werden bei aktiviertem Word-Wrapping in der Anzeige umgebrochen.
Am rechten Bildschirmrand werden die Elemente innerhalb des aktuellen Knotens angezeigt, sodass diese zum Bearbeiten und Duplizieren sofort zugänglich sind. An dieser Stelle erwartet sich der geübte Bearbeiter jedoch eine Anzeige aller im aktuellen Kontext infrage kommenden Tags. Da der Editor durchaus in der Lage ist, die TEI-Kodierung

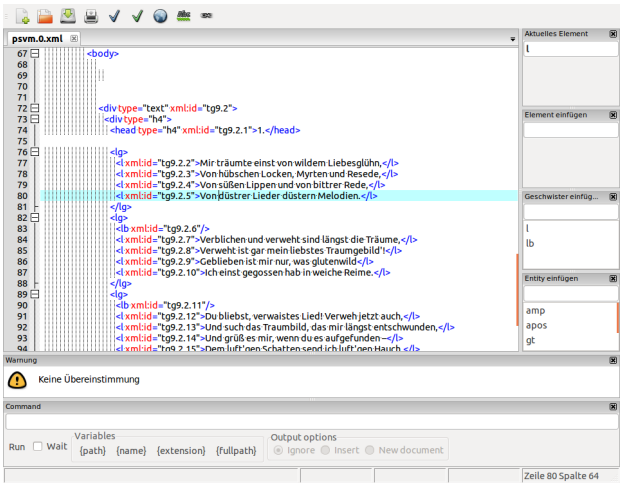

Abbildung 4: Die Anzeige von Warnungen und die Möglichkeit, externe Programme aus dem XML Copy Editor zu starten.

zu validieren, wäre eine entsprechende Anzeige hier wohl machbar.
Die Entscheidung, ob XML-Tags und/oder ihre Attribute überhaupt angezeigt werden, ist vom Bearbeiter einstellbar. Dieser Umstand mag verwundern. Er ist jedoch immer dann sehr hilfreich, wenn umfangreiche Texte bearbeitet werden müssen und der Bearbeiter seine Konzentration gerade auf die Inhalte und nicht auf deren Kodierung richtet.

Zudem beherrscht der Editor die Funktion, dass Tags „gesperrt" werden können. Damit sind diese vom (versehentlichen) Verändern ausgenommen. Diese Funktion wird zudem dann aktiv, wenn das Ausblenden der Tags angewählt wurde. Abbildung 3 zeigt den Bearbeitungsmodus mit ausgeblendeten und gesperrten XML-Tags.
Die verwendeten Bildschirmfonts, sowie ihre Darstellungsgröße ist veränderbar.

Die Highlights des Editors sind in den Menüs gut untergebracht:

- Neben der Möglichkeit TEI-kodierte Texte zu validieren und sie auf ihre Wohlgeformtheit zu überprüfen, ist der Editor in der Lage, innerhalb des zu bearbeitenden Dokuments XPath-Ausdrücke anzuwenden, mit denen ein Dokument anhand seiner XML-Struktur durchsucht werden kann.
- Die Angabe eines externen XSL-Stylesheets zur Umwandlung (Transformation) des Dokuments in einen bestimmten Dokumententyp via XSLT ist ebenso möglich, wie die Umwandlung durch ein in das Dokument eingebettetes Stylesheet. Erstes ist im Menü unter „XML" zu finden, zweiteres funktioniert über das entsprechende Icon in der Toolbar. Diese komfortable Möglichkeit, XSLT-Prozesse unterschiedlich anzusprechen, bieten nur wenige Editoren.
- Die Suchfunktionen bieten die Möglichkeit, reguläre Ausdrücke anzuwenden. Leider ist dies im Zusammenhang von „Suchen und Ersetzen" nicht möglich.
- Die Funktion „Öffne großes Dokument" deaktiviert die Möglichkeiten des Auf- und Zuklappens von Dokumententeilen sowie das farbliche Hervorheben der Syntax. Damit reagiert der Editor im Umgang mit großen Dokumenten deutlich schneller.

Vorteile
+ komfortable Anzeigemöglichkeiten
+ einfache und übersichtliche Bedienung
+ einfache Vorschlagsfunktion für Tags und Attribute
+ XML-Tags können vor Veränderung geschützt werden
+ reguläre Ausdrücke beim Suchen
+ XPath und XSLT-Unterstützung
+ XML-TEI-Validierung
+ OpenSource

Nachteile
- defekte Installationspakete
- keine vollständige TEI-Unterstützung
- keine Template-Verwaltung
- wenig Möglichkeiten, externe Programme einzubinden

Abbildung 4 zeigt, dass der Editor Meldungen in einem Frame am unteren Bildschirmrand ausgibt. Ebenso klappt an dieser Stelle eine Eingabezeile aus, über die externe Programme (wie z. B. ein CVS/SVN/Git-Client) aufgerufen werden können.

Das TextGritLab

Aus dem von der deutschen Forschungsgemeinschaft (DFG) finanzierten Projekt TextGrid (= Virtuelle Forschungsumgebung für die Geisteswissenschaften) ist das TextGridLab hervorgegangen [11]. Dieses stellt als OpenSource-Software nicht nur eine vollständige Arbeitsumgebung für den Client zur Verfügung, sondern bietet ernsthaft Interessierten zudem die Möglichkeit, die erstellten Texte innerhalb eines Grid-Verbundes dauerhaft abzulegen. Über diesen Verbund freigegebene Texte stehen automatisch online unter einer Creative Commons Lizenz zur Verfügung.

Die Installation des TextGridLab geschieht denkbar einfach und ist unter [11] bestens dokumentiert.
Mit …

```
wget http://www.textgridlab.org/download/2.0.4/TextGridLab-2.0.4-linux.gtk.x86.zip
```
bzw.
```
wget http://www.textgridlab.org/download/2.0.4/TextGridLab-2.0.4-linux.gtk.x86_64.zip
```

… holt man sich die notwendigen Dateien und entpackt diese in seinem Arbeitsverzeichnis via

```
unzip TextGridLab-2.0.4-linux.gtk.x86.zip
```
bzw.
```
unzip TextGridLab-2.0.4-linux.gtk.x86_64.zip
```

Abbildung 5: Der Startbildschirm des TextGridLab.

Auf Ubuntu-Systemen muss zudem darauf geachtet werden, dass die Programmbibliothek libwebkitgtk-1.0-0 installiert ist.

```
sudo apt-get install libwebkitgtk-1.0-0
```

zeigt an, ob eine Installation notwendig ist und installiert diese.
Der Startbildschirm von TextGridLab in Abbildung 5 lässt bereits erahnen, dass dieses Werkzeug mit deutlich mehr als einem XML-Editor aufwartet:

- Die Suche führt direkt in das TextGrid-System, über das alle dort abgelegten und indizierten Texte erreichbar sind.
- Die Projektverwaltung ermöglicht einem Redakteur und/oder Herausgeber umfangreiche Projekte zu steuern.
- Für die Text-Bild-Verwaltung wird an dieser Stelle auf den eingebauten und speziell ausgerichtete Editor hingewiesen.
- Daten (d. h. Texte, Bilder, Metadaten etc.), die als solche zusammengehörig sind, können über den Einstieg „Aggregationen“ mittels einfachem Drag & Drop bearbeitet bzw. zusammengeführt werden.
- Das „Trierer Wörterbuchnetz“ wurde in das TextGrid integriert und ist über die Wörterbuchsuche erreichbar.

Die wichtigsten Funktionalitäten, die am Startbildschirm genannt sind, könne unter [12] in einer Kurzbeschreibung nachgelesen werden.
Das TextGridLab wurde auf der Basis von Eclipse entwickelt. Jene Bearbeiter, die mit dieser Entwicklungsumgebung bereits Erfahrung gesammelt haben, werden sich schnell zurecht finden.
Allen anderen wird der Einstieg erleichtert, indem über das Menü für verschiedene Aufgabenstellungen vordefinierte Ansichten (= „Perspektiven“) aktiviert werden können (Abbildung 6).

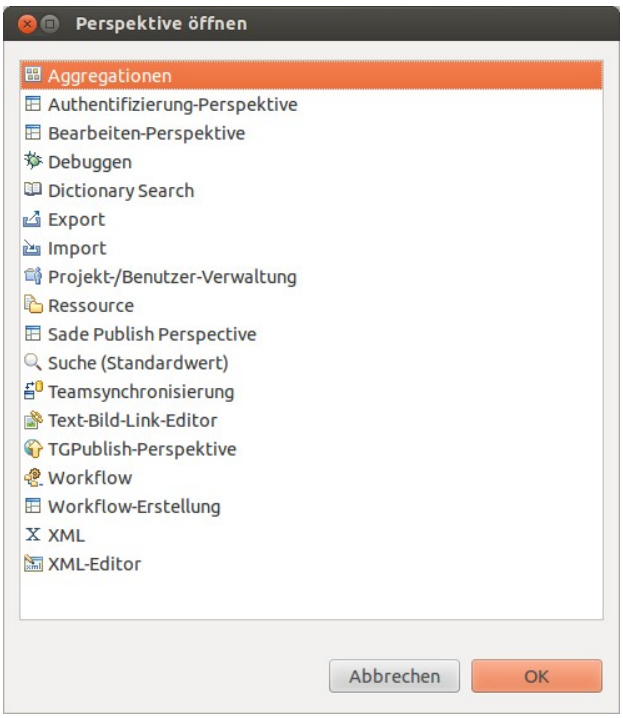

Abbildung 6: Die Anzeige der Arbeitsumgebung kann einfach über vordefinierte „Perspektiven“ in verschiedene Modi umgeschaltet werden.

Ein großer Vorteil ergibt sich aus der Eclipse-Basis: Die Entwickler müssen nicht jedes Feature selbst neu programmieren, sondern können auf die Vielfalt der bereits vorhandenen Plugins für Eclipse zurückgreifen. Jene, die von den Machern des TextGridLab freigegeben wurden, sind über den „Marketplace“ einfach nachzuinstallieren.

Die Vielfalt dabei reicht von einfachen Möglichkeiten des Umbruchs langer Zeilen in der Anzeige bis hin zu Konnektoren für Publikationsserver wie Sade.
Hat man sich einmal an die geteilte Ansicht innerhalb von Frames gewöhnt, zeigt sich die Arbeitsumgebung von Ihrer besten Seite. Ein geöffnetes Dokument in der Ansicht des XML-Editors wird in einem elegant zurückhaltenden Fenstermodus angezeigt. Abbildung 7 zeigt die typische Dreiteilung des Bildschirms in die XML-Baumstruktur, die mit XML-Tags beschriebenen Inhalte sowie die Eigenschaften und Werte einer aktuell angewählten Kategorie. Auf kleineren Bildschirmen können bzw. müssen einzelne Frames für ein übersichtliches Arbeiten geschlossen werden.

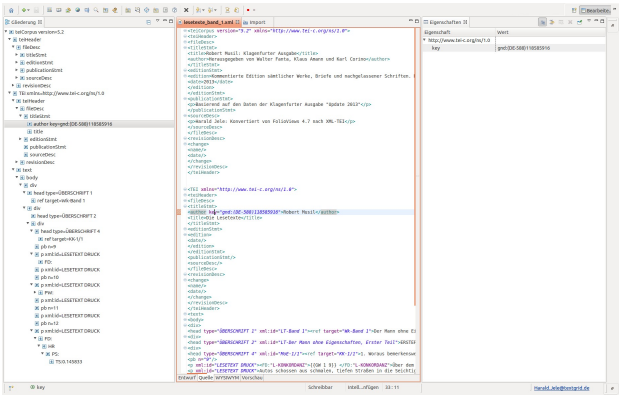

Abbildung 7: Eine geöffnete Datei in der Perspektive „Bearbeiten“.

Der volle Funktionsumfang des TextGridLab erschließt sich einem Bearbeiter jedoch nur im Zusammenspiel mit einer Anmeldung am TextGrid-System. Leider sind von diesem Umstand viele der Kernfunktionen betroffen:

- die Projektverwaltung,
- die Benutzerverwaltung,
- Dateioperationen wie „als Revision speichern“,
- die Gestaltung eines Workflows.

Als ein weiterer Nachteil zeigt sich, dass die Programmierer viele fertige Eclipse-Plugins, die in der täglichen Arbeit eine zentrale Rolle spielen, nicht über den Marketplace freigegeben haben. Dazu zählt z. B. auch die Integration eines CVS/SVN/Git-Clients. Ein solcher ist als Plugin natürlich vorhanden. Die TextGridLab-Entwickler haben jedoch leider keinen über ihren Marketplace im Angebot.
Beim Import von Texten ins TextGrid-System werden diese automatisch auf Wohlgeformtheit und TEI-Validität geprüft. Nur jene Texte, die beiden Kriterien gehorchen, werden im Anschluss in einen Volltextindex aufgenommen und sind darüber auffindbar. Alle anderen werden nur gespeichert.

Vorteile
+ hohe Flexibilität durch die Eclipse-Basis
+ sehr übersichtliche Arbeitsumgebung
+ viele Funktionen für das Editieren von TEI-kodierten Texten vorhanden
+ integrierte Updatefunktion
+ komfortable Textansichten
+ gute Integration der TEI-Vorgaben
+ OpenSource

Nachteile
- keine ausgereifte Vorschlagsfunktion für Tags und Attribute
- keine Projektverwaltung im lokalen Dateisystem
- keine lokale Benutzerverwaltung
- kein Aufruf eines externen XSL-Stylesheets möglich
- sehr viele nützliche Exclipse-Plugins bleiben im Marketplace unbeachtet

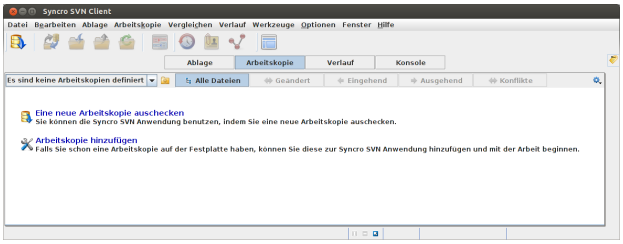

Abbildung 8: Der integrierte SVN-Client bietet im <oXygen/> XML Editor komfortable Funktionen zur Versionsverwaltung.

<oXygen/> XML Editor

Der Erfolg vieler umfangreicher Editionsprojekte beruht darauf, dass als zentrales Werkzeug der <oXygen/> XML Editor eingesetzt wird.
Entsprechend der Eigenwerbung versucht der Hersteller mit diesem gleichzeitig sowohl Anfänger als auch sehr fortgeschrittene Bearbeiter anzusprechen [13]. Um diesen Spagat zu überbrücken, unternimmt der Hersteller auch jede Mühe, um einerseits sämtliche XML-Methoden und -Technologien in sein Produkt zu integrieren und andererseits diese so (in einer geschickt gestalteten Menü- und Kontextmenüstruktur) zu platzieren, dass auf diese zwar bei Bedarf rasch zugegriffen werden kann, ein Anfänger ob der Funktionsfülle jedoch nicht gleich überfordert wird.
<oXygen/> ist im Gegensatz zu den beiden anderen Editoren nicht OpenSource. Eine kostenlose und 30 Tage funktionsfähige Version kann jedoch gefahrlos installiert werden. Da der Editor auch als „portable“ Software angeboten wird, verändert eine Installation das Betriebssystem nicht.
Wer mit dem TextGridLab bereits vertraut ist, dem kommt Einiges an der Bedienung bekannt vor: Wie dieses ist auch der <oXygen/> XML Editor als Plugin zu Eclipse realisiert und profitiert so vom flexiblen Unterbau.
Während im TextGridLab der ambitionierte XML-Bearbeiter leider angehalten ist, sämtliche fehlende Eclipse-Plugins selbst nachzuinstallieren und auf ihre Funktionsfähigkeit zu überprüfen, sind diese hier vorbildlich eingebettet. Dies gilt für Funktionen zur Versionierung (Abbildung 8) ebenso wie z. B. für den Umbruch überlanger Zeilen in der Anzeige.
Abbildung 9 zeigt die typische Fensterteilung im Editiermodus, die bei zugeklapptem Projektfenster der Anzeige von TextGridLab sehr ähnlich ist. Für den Bearbeiter, der zur Kontrolle der Funktionalität einzelner XML-Strukturen zwischendurch eine Textpräsentation in einem anderen Anzeigeformat wie HTML, PDF oder e-Pub wünscht, ist ein schneller Zugriff auf die entsprechenden Stylesheets gegeben (ohne, dass deren Pfad im Dokument eingebettet werden muss).

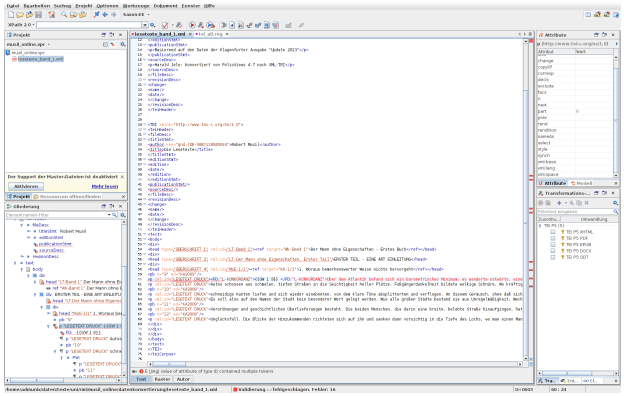

Abbildung 9: Eine geöffnete Datei im <oXygen/> XML Editor.

Die Vorgaben des TEI-Konsortiums sind in diesem Editor vorbildlich integriert. Davon profitieren nicht nur Funktionen zur Überprüfung der Validität eines Dokuments ebenso wie das Debugging, sondern auch die Vorschlagsfunktionen, die einem Bearbeiter bei der Schachtelung der XML-Tags sämtliche zulässigen Möglichkeiten anbietet.
Da im Editor Konnektoren zu allen gängigen XML-Datenbanken realisiert sind, ist eine Anbindung der meisten Publikationsserver, die XML-TEI-kodierte Texte verarbeiten können, möglich.
Sämtliche Vorgänge der Projektverwaltung sind im Dateisystem abgebildet. Daher ist den Projektverantwortlichen mit einfachen Mitteln möglich, diese nach ihren Bedürfnissen einzurichten.
Die Template-Verwaltung erlaubt während dem Editieren, Textteile als Templates zu definieren und als solche aus einer Liste jederzeit aufzurufen und wiederzuverwenden.
Sämtliche XML-basierte Methoden, mit denen Dokumente günstiger Weise weiterverarbeitet werden, sind im <oXygen/> XML Editor integriert. Dazu zählen neben dem bereits erwähnten XSLT auch XQuery, XPath und das moderne XProc, bei dem eine Stapelverarbeitung von XML-Dokumenten durch die Definierung sogenannter Pipelines erfolgt.

Vorteile
+ ausgereifte XML-Arbeitsumgebung
+ ausgezeichnete TEI-Unterstützung
+ Anbindung einer Vielzahl an XML-Datenbanken
+ komfortables Debugging

Nachteile
- kein OpenSource-Produkt
- ressourenhungrig
- unterstützt nur Oracle-Java

Fazit

Um eine vollständige XML-TEI-Arbeitsumgebung am Client zusammenzustellen reichen im Grunde sämtliche Tools, die eine der gängigen Linux-Distributionen bereitstellt: Eine geschickt gewählte Kombination aus XML Copy Editor, dem Lieblings-CVS/SVN/Git-Client, sowie einem der gängigen OpenSource-Tools zur Projektverwaltung bieten einem Bearbeiter als auch einem Redakteur schon eine Menge an Arbeitserleichterung.
Eine deutliche Steigerung der Möglichkeiten und vor allem des Komforts ist gegenüber einem solchen Baukastensystem im Einsatz von TextGridLab zu erkennen. Insbesondere dann, wenn der Bearbeiter seine Texte im TextGrid-System ablegt, lassen sich in dieser integrierten Arbeitsumgebung bereits sehr viele Funktionen vom Editieren, über die Projektverwaltung bis hin zum Publizieren unter einer Haube abbilden. Zum vollständigen Glück fehlt jedoch noch das Eine oder Andere. TextGridLab wird jedoch kontinuierlich weiterentwickelt. Man kann also gespannt sein, ab wann man auch umfangreiche Projekte allein damit abwickeln kann.
<oXygen/> XML Editor bringt sämtliche Funktionen mit, die die beiden zuvor besprochenen auszeichnen und erweitert den gelieferten Umfang zudem noch deutlich.
Wer sich ernsthaft mit XML-TEI-kodierten Dokumenten auseinanderzusetzen hat und den Komfort des Editors erkannt hat, wird auf Dauer an diesem kaum vorbeikommen.
Eine OpenSource-Variante des Editors wäre wünschenswert, auch mit dem Preis, dass eine solche eventuell nicht den vollen Funktionsumfang bietet. Die Vielzahl an OpenSource-Technologien, die in den <oXygen/> XML Editor bereits integriert sind, lassen einen solchen Wunsch als durchaus gerechtfertigt erscheinen.

Quellen

[1] TEI: Text Encoding Initiative
[http://www.tei-c.org]

[2] Textkonvertierung via „OxGarage“
[http://www.tei-c.org/oxgarage/]

[3] Public Stylesheets of the TEI consortium
[https://github.com/TEIC/Stylesheets]

[4] Saxon: The XSLT und XQuery Processor
[http://saxon.sourceforge.net]

[5] Sade: Scalable Architecture for Digital Editions
[http://www.bbaw.de/telota/software/sade/sade-1]

[6] XTF: eXtensible Text Framework
[http://xtf.cdlib.org]

[7] Comparison of XML editors
[http://en.wikipedia.org/wiki/Comparison_of_XML_editors]

[8] XML Copy Editor
[http://xml-copy-editor.sourceforge.net]

[9] Conglomerate: XML For Everyone
[http://www.conglomerate.org]

[10] XML Copy Editor kompilieren
[http://sourceforge.net/p/xml-copy-editor/svn/135/tree/INSTALL]

[11] Das TextGridLab
[http://textgridlab.org]

[12] Kurzbeschreibung der Editoren und Werkzeuge sowie Dienste und Features im TextGrid-Lab
[http://www.textgrid.de/ueber-textgrid/tools-services-ressourcen/]

[13] Der <oXygen/> XML editor
[http://www.oxygenxml.com]

Der Autor

Dr. Harald Jele ist Mitarbeiter an der Universität Klagenfurt und beschäftigt sich zur Zeit (auch) mit der Migration der Klagenfurter Ausgabe Robert Musils.