Report über die AP6-Entwicklung des vergangenen Jahres (R 6.0.3)

Version 2012-10-31
Arbeitspaket 6
verantwortlicher Partner Universität Würzburg

TextGrid

Vernetzte Forschungsumgebung in den eHumanities

Ein Projekt im D-GRID

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

Projekt: TextGrid - Vernetzte Forschungsumgebung in den eHumanities
BMBF Förderkennzeichen: 01UG0901A
Laufzeit: Juni 2009 bis Mai 2012

Dokumentstatus: <Entwurf, Final>
Verfügbarkeit: <öffentlich, TextGrid-intern>
Autoren:

Thorsten Vitt, UWÜ
Toolverantwortliche

Revisionsverlauf:

Datum	Autor	Kommentare
31.10.2012	Thorsten Vitt	Einleitung, Überarbeitung

Inhaltsverzeichnis

1 Einleitung

Zum Digital-Humanities-Festakt am 12. Juli 2011 wurde die stabile Produktivversion 1.0 des TextGridLab veröffentlicht. In dieser Version waren im Wesentlichen Tools vorhanden, die zu diesem Zeitpunkt als stabil galten und die für die Benutzung des TextGridLab im Kern notwendig waren.

Die Arbeiten im abschließenden Jahr des Projektes umfassten dann im Wesentlichen vier Ziele:

1. Die (Weiter-)entwicklung der übrigen Tools und ihre Integration in das TextGridLab
2. Die weitere Modularisierung des TextGridLab
3. Die Einarbeitung von Feedback zu Version 1.0 in die entsprechenden Tools
4. Eine Internationalisierung, zunächst mit deutsch- und englischsprachigen Texten in der Benutzungsschnittstelle. Dies erforderte Anpassungen in allen Tools des TextGridLab.

Zum Abschluss des Projekts konnte schließlich eine Version 2.0 veröffentlicht werden, die nicht nur aktualisierte Versionen der Tools aus TextGridLab 1.0 enthält und die zusätzlichen Werkzeuge zur Nachinstallation anbietet, sondern durch ihre Modularität und die eingebauten Update-Mechanismen eine gute Grundlage für eine dezentrale weitere Pflege und Erweiterung des Ökosystems TextGrid bildet.

Dieser Bericht enthält eine Beschreibung dieser Neuausrichtung sowie Kurzbeschreibungen zu den für die einzelnen Tools, soweit sie in den Verantwortungsbereich von AP6 fallen, erfolgten Entwicklungen im Berichtszeitraum.

Von der ursprünglich vorgesehenen Form als Endnutzerdokumentation wurde Abstand genommen: Es hat sich als bessere, d.h. für die Endnutzer verständlichere Lösung erwiesen, diese Dokumentation von Nichtentwicklern (auf der Basis von Stichpunkten und mit Korrekturphase durch die Entwickler) schreiben zu lassen. Die Dokumentation wurde in AP4 als Handbuch zusammen mit Version 2.0 veröffentlicht, ist als Online-Hilfe integriert und wird auf https://dev2.dariah.eu/wiki/display/TextGrid/User+Manual+2.0 stets aktuell gehalten.

2 Modularisierung und Updatemechanismen

Aus technischer Perspektive war das TextGridLab schon immer zu einem gewissen Grade modular: Es ist, wie bei Eclipse-basierten Anwendungen üblich, aus einer Reihe von *Plugins* aufgebaut, die aus Eigenentwicklungen, der Eclipse-Plattform und von dritten Open-Source-Anbietern stammen und über Programmierschnittstellen miteinander interagieren.

Die Zusammenstellung dieser Plugins zu einem fertigen Produkt erfolgte allerdings in Version 1.0 noch bereits zur Compilezeit, sodass Endnutzer das TextGridLab als ein monolithisches Programm erhielten, das im Ganzen heruntergeladen und entpackt wurde, und das zum Update vollständig durch eine neue Version ersetzt werden musste.

Für TextGridLab 2.0 mit seinen Tools aus recht unterschiedlichen Fachcommunities und mit unterschiedlichem Entwicklungsstand erwies sich dies als zu statisch. Der Code wurde deshalb so restrukturiert, dass er sich für die Nutzung der auch aus Eclipse bekannten Update- und Installationsmechanismen (*p2*, *Marketplace*) eignet.

2.1 Komponenten

Das TextGridLab wurde dabei in die folgenden Komponenten zerlegt:

- *textgridlab-dependencies* enthält Bestandteile, die im Lab verwendet werden, jedoch nicht spezifisch für die Verwendung im TextGridLab- bzw. Eclipse-Kontext sind. Diese Komponenten werden hier im wesentlichen aggregiert und in Features verpackt, um als Bestandteile des Lab direkt nutzbar zu werden. Integriert sind sowohl Bibliotheken aus eigener wie fremder Produktion als auch Client-Bibliotheken für die Webservices des TextGridRep.

- *core* enthält jene Komponenten des TextGridLab, die zum Betrieb notwendig und von anderen Komponenten benötigt werden, etwa die GUI-Grundstruktur und die Lab-seitige Kapselung und Modellierung des TextGridRep.

- *base* enthält diejenigen Bestandteile, die zwar den Kern der allgemeinen TextGridLab-Funktionalität bilden, aber nicht API sind und darum von anderen Komponenten nicht direkt angesprochen werden. Dazu gehören z.B. Willkommensbildschirm, Workflow-editor, …

  In dieser Komponente wird zudem das downloadbare TextGridLab gebaut. Entsprechend enthält die Base-Komponente Abhängigkeiten zu allen Tools, die das herunterladbare Produkt ausmachen, d.h. mit *base* kommen die Komponenten *core, help, xmleditor* und *linkeditor* automatisch mit.

- *help* enthält die Online-Hilfe.

- *xmleditor* enthält den eingebauten XML-Editor.

- *linkeditor* enthält den Text-Bild-Linkeditor.

Neben diesen Kernbestandteilen gibt es eigene Komponenten für die nachinstallierbaren Tools:

- *collatex* für den Kollationierer und sein User Interface,
- *dictionaries* für das Frontend zum Wörterbuchnetz und den Wörterbuchlinkeditor,
- *digilib* enthält die Anbindung des Viewers für große Bilddaten, Digilib, ans TextGrid-Lab,
- *glosses* mit dem Glosseneditor,
- *linguistics* enthält die Labkomponenten der linguistischen Tools: Lemmatizer, COSMAS, LEXUS und ANNEX;
- *noteeditor* für den Noteneditor MEISE,
- *sadepublish* enthält die Web-Preview-Komponente, die nach SADE publiziert,
- *ttle* für die Labanbindung des Text-Text-Linkeditors.

(die übrigen TextGrid-Tools stehen als reine Webservices über das Workflowtool zur Verfügung.)

2.2 Technische Struktur der Komponenten

Jede Komponente besteht aus Plugins und wenigstens einem Feature:

- *Plugins* enthalten weiterhin wie in der nicht modularisierten Version den eigentlichen Code und die Ressourcen des Projekts. Dabei wurden jedoch nach Möglichkeit Bibliotheken von Dritten, Service-Clients (aus generiertem Code) sowie TextGrid-Bibliotheken (wie der Link Rewriter), die auch außerhalb des Lab Verwendung finden, aus den eigentlichen Lab-Plugins herausgelöst. Stattdessen werden diese Bibliotheken in einem separaten, automatisierten Prozess (Modul *textgridlab-dependencies*) in Eclipse-Plugins gewandelt (soweit nötig) und in einem *p2*-Repository zur Verfügung gestellt, wo sie dann sowohl von Entwicklern in Eclipse als auch vom das TextGridLab bzw. dessen einzelne Komponenten erzeugenden Buildprozess genutzt werden können.

  Dieser Schritt hilft, Redundanzen in Form mehrfach eingebundener Bibliotheken zu vermeiden, Abhängigkeiten zwischen Labplugins, die sich eigentlich auf die eingebetteten Bibliotheken bezogen, zu entschärfen sowie Codegenerierungsschritte, die zuvor bei *jedem* Labentwickler lokal abliefen, zu zentralisieren.

- *Features* bündeln eine Reihe von Plugins, zudem können sie Abhängigkeiten zu anderen Features enthalten. Sie sind derjenige Bestandteil, der praktisch zur Installation angeboten wird. Bei der Installation eines Features werden all seine Plugins sowie alle Features, von denen das zu installierende abhängt, mitinstalliert.

Die Bestandteile einer Komponente werden gemeinsam compiliert und unterliegen damit einem gemeinsamen Release-Zyklus.

2.3 Update Sites

Zu jeder Komponente gehört dabei eine *Update Site*, die die entsprechenden Plugins und Features anbietet. Jede Update Site enthält einen Metadatenkatalog einschließlich der Versions- und Abhängigkeitsmetadaten, der von den im TextGridLab integrierten Installationstools ausgewertet werden kann. Diese toolspezifischen Update Sites werden zu drei größeren, zusammengesetzten Sites aggregiert: Eine *stabile* Site für stabile, getestete Releases entsprechender Tools, eine *Beta*-Site für ausgesuchte, interne getestete Releases von Tools, die Beta-Qualität haben, sowie eine *Nightly*-Site für den aktuellsten, automatisch übersetzten Stand der einzelnen Tools.

Es ist im Allgemeinen möglich, Tools aus unterschiedlichen dieser Quellen zu mischen, also z.B. mit einer stabilen Version des Lab zu arbeiten, dahinein aber das Beta-Tool CollateX zu installieren und den XML-Editor auf die Nightly-Version mit den neuesten Funktionalitäten zu aktualisieren.

2.4 Automatischer Updatemechanismus

Das TextGridLab prüft nach dem Start im Hintergrund, ob es noch aktuell ist. Dazu wird für das Produkt selbst und für alle nachinstallierten oder aktualisierten Tools geprüft, ob es auf den entsprechenden Update Sites eine neuere Version gibt. Falls ja, informiert das Lab den Benutzer über eine unaufdringliche Benachrichtigung, aus der heraus per Klick unmittelbar das Update eingeleitet werden kann.

Der Mechanismus ist konfigurierbar: Benutzer, die die automatischen Checks seltener oder gar nicht wünschen, können sie abstellen und bei Bedarf manuell über einen Menüpunkt auslösen, und Benutzer werden in jedem Falle gefragt, bevor irgendetwas installiert wird.

Andererseits erlaubt der Mechanismus es den Benutzern, ihr Lab aktuell zu halten, und uns, Bugfixes und sonstige Aktualisierungen rasch an alle Benutzer zu verteilen.

2.5 Marketplace

Die Nachinstallation der nicht mitgelieferten Tools wird über eine GUI vereinfacht, für die die Eclipse-Komponente *Marketplace Client* nachgenutzt wurde: Benutzer können in einer Übersicht die gewünschten Tools auswählen, die Software übernimmt das Hinzufügen der Update Site und die Installation. Über den Marketplace können auch externe Tools angeboten werden. Dies wurde etwa mit dem kommerziellen, in der Community recht beliebten XML-Editor *oXygen XML Editor* realisiert, dessen Plugin-Variante über den Marketplace in TextGridLab installiert werden kann (eine entsprechende Lizenz dafür müssten die Nutzer allerdings separat erwerben).

Hierbei wurde zunächst ein recht einfacher Open-Source-Marketplace-Katalog[1] installiert, ein Ausbau kann später erfolgen.

[1] http://sourceforge.net/p/marketplace-cat/home/Home/

2.6 Build

Für den Build wurde von der noch für Version 1.0 verwendeten Lösung aus dem Eclipse-PDE-Build in Verbindung mit eigenen Skripten auf eine Lösung auf der Basis von Maven und dem darauf aufbauenden *Tycho*[2] gesetzt. Dieses Tool hat in letzter Zeit einen beträchtlichen Entwicklungsschub vollzogen und wird nun bereits von zahlreichen Eclipse-Projekten eingesetzt; zu den Vorteilen gegenüber dem bisherigen Build gehört neben der besseren Modularisierbarkeit und der aktiveren Community eine bessere Unterstützung durch Tools wie etwa den Integration-Build-Server Jenkins[3] und ein leichteres Setup der Buildumgebung: Übersetzt werden kann das Lab bzw. einzelne Komponenten auf jedem Rechner, auf dem Maven 3 läuft, eine separate Installation von Eclipse etc. ist nicht nötig. Für die Komponenten wurde eine recht einheitliche Projektstruktur umgesetzt, die Ende 2012 zu einer dokumentierten Vorlage ausgebaut wird.

2.7 TextGridLab SDK

Neben den ins Endnutzer-TextGridLab installierbaren Features werden durch den Build ebenfalls *Software-Development-Kit*-Features für die Basiskomponenten erstellt, die es ermöglichen, Tools für das TextGridLab zu entwickeln, ohne dabei den kompletten TextGridLab-Sourcecode in den eigenen Eclipse-Workspace auszuchecken und lokal zu compilieren. Stattdessen kann die entsprechende TextGridLab-Version Teil der Target Platform werden und im Workspace findet sich nur der Quellcode des eigenen Tools.[4]

[2] http://www.eclipse.org/tycho
[3] http://www.jenkins-ci.org/
[4] Eine detaillierte Anleitung dazu gibt es im öffentlichen Wiki unter https://dev2.dariah.eu/wiki/display/TextGrid/TextGridLab+Development+Environment

3 AP 6.1

3.1 Metadaten – Editor/ - Annotationen

Der Metadateneditor hatte zu Version 1.0 durch die komplette Überarbeitung des TextGrid-Metadatenmodells eine grundlegende Überarbeitung erfahren.

Im letzten Jahr des Projekts erfuhr der Metadateneditor vor allem Bugfixes und Usability-Anpassungen, die aus Feedback zu Version 1.0 resultierten. Dazu gehörte eine verbesserte Darstellung des Zusammenhangs zwischen den unterschiedlichen Objekttypen, Verbesserungen bei der Integration mit anderen Tools wie Aggregationseditor und Import, Verbesserung bei Autocompletion bzw. Normdateianbindung und Verbesserungen im generierten TEI-Header.

3.2 XML-Editor

Der eingebaute XML-Editor erfuhr eine Reihe von Bugfixes als Ergebnis aus Feedback zu TextGridLab 1.0 und Nutzerprojekten, unter anderem im Umgang mit CSS-Stylesheets des Benutzers für die Textansicht, im Umgang mit DTDs und zur Bearbeitung aller auf XML basierender Formate des TextGridLab.

Auf Anregung von Nutzern wurde zudem eine zusätzliche Ansicht geschaffen, in der schnell eine mittels vom User angegebenem XSLT-Stylesheet transformierte Darstellung in einer Browseransicht gezeigt wird: damit können die Benutzer etwa prüfen, ob ihr Dokument auch in der später avisierten Webdarstellung zufriedenstellend aussieht.

3.3 Link-Editor Text

Der Link-Editor Text dient als Eingabehilfe für Links in XML-Dateien. Benutzer haben die Möglichkeit, Verknüpfungen zwischen beliebigen Elementen von zumindest für den Benutzer lesbaren TextGrid-Dokumenten in einer TTLE - TEI-Datei zu erstellen. Darüber hinaus wird der Link-Editor Text zur Validierung bereits existierender Verknüpfungen eingesetzt.

Nachdem im vergangenen Jahr ein ausführliches Pflichtenheft in Zusammenarbeit mit an der Funktionalität interessierten Projekten erstellt wurde (vgl. Report 6.0.2), wurde in diesem Jahr die Grundfunktionalität des Editors zum Verlinken von XML-Dokumenten und der Bearbeitung von Linkeigenschaften implementiert. Dabei wurde mit dem Ziel einer Integration auch in künftige Weboberflächen eine hybrider UI-Ansatz aus Lab- und Web-Oberfläche gewählt.

Der Linkeditor Text steht mit Version 2.0 als Betaversion über den Marketplace zur Verfügung.

3.4 Lemmatisierer

Der Lemmatisierer wurde im Rahmen der Komponente mit linguistischen Tools in TextGridLab Version 2.0 eingebunden. Neben Bugfixes wurde er konfigurierbarer gemacht und besser an die TextGrid-Infrastruktur angebunden.

3.5 Streaming-Editor

Als Streaming-Editor wurde ein XSLT-2.0-Prozessor um die Möglichkeit, TextGrid-URIs aufzulösen, erweitert und in einen Web-Service gekapselt. Er steht nun über das in TextGrid-Lab 2.0 integrierte Workflowtool zur Verfügung.

3.6 Import-/Export-Tool (TextGridLab)

Im TextGridLab-Importtool – neu in TextGridLab 1.0 – wurden diverse Bugs insbesondere in selteneren Anwendungsszenarien behoben, die in den Tests für 1.0 nicht aufgefallen waren. Zudem gab es Usability-Verbesserungen: Insbesondere wird beim Im-/Export defekter XML-Dateien, bei denen das Link-Rewriting scheitert, nun ein Import ohne Rewriting vorgenommen und eine Warnmeldung ausgegeben statt den Import der Datei erfolglos abzubrechen. Der Link-Rewriting-Mechanismus wurde zudem mit neuen Standardtypen versehen und robuster gegen bestimmte XML-Probleme (nicht auflösende DTDs) gemacht.

3.7 Tokenizer

Der Tokenizer wurde durch eine auf der Plattform für linguistische Tools GATE basierende Webservice-Version ausgetauscht. Er ist über das in TextGridLab 2.0 integrierte Workflowtool ansprechbar.

3.8 SADE-Publisher (zuvor Web-Preview, Web-Publisher)

Inhalte aus dem TextGridLab, die dauerhaft publiziert werden, erscheinen im TextGridRep-Portal[5]. Da dort jedoch *alle* Texte erscheinen, sind die Such- und Anzeigemöglichkeiten eher darauf ausgelegt, für alle publizierten Texte zu passen, als projektspezifische Such- und Visualisierungsmöglichkeiten anzubieten.

Für solche projektspezifischen Webpublikationen ist individuelle Anpassungs- und Programmierarbeit nötig. Einen oftmals geeigneten Ausgangspunkt dafür bietet das an der Berlin-Brandenburgischen Akademie der Wissenschaften entwickelte Paket *SADE*, das einen Application Server mit der XML-Datenbank eXist und mit Digilib zu einer fertigen Appliance verbindet, die bereits out of the box Grundfunktionen zur Anzeige und Suche in XML-Dateien mitbringt und die Ausgangsbasis für eigene Webpublikationen bilden kann.

Der für TextGridLab 2.0 über den Marketplace angebotene SADE-Publisher bietet eine einfache Integration in das Lab: Mit ihm können direkt aus dem TextGridLab per Drag & Drop TextGrid-Daten in eine entsprechend angepasste SADE-Instanz publiziert werden.

3.9 Vorbereitung der Integration von Drittanwendungen

Durch die im ersten Teil beschriebene Modularisierung und Integration des Eclipse-Updatemechanismus (p2) ist es grundsätzlich möglich, auch Dritttools, die ohne Kenntnis von TextGrid-APIs entwickelt wurden, in das TextGridLab zu integrieren – solange es sich um Eclipse-Plugins handelt. Dies funktioniert auch deshalb, weil für zahlreiche Operationen (Kapselung des Ressourcenzugriffs, Grundaufbau der GUI, Auswahl der Editoren …) Kompatibilität zu Eclipse-APIs gewahrt wurde.

[5] http://www.textgridrep.de/

Eine zusätzliche Komponente adaptiert die Standard-Java-URL-Handler auf den entsprechenden Mechanismus zum Zugriff auf TextGrid-Objekte im Lab. Damit können Tools, die Daten über URLs laden (und etwa http://-Adressen auflösen können), auch mit textgrid:-URIs umgehen.

Nützlich ist das zum Beispiel bei der Integration des kommerziellen XML-Editors oXygen: Dieser kann in das Lab installiert werden und auch z.B. XML-Dokumente gegen in TextGrid gespeicherte Schemata validieren.

4 AP 6.2

4.1 Text-Bild-Link-Editor

Der Text-Bild-Linkeditor war bereits Bestandteil der stabilen TextGridLab-Version 1.0.

Im abschließenden Jahr des Projektes wurden hier im wesentlichen Bugs repariert, etwa im Umgang mit bestimmten Bildern und in der Integration mit dem Rest des TextGridLab, sowie Usability-Verbesserungen auf Anregung von Nutzern durchgeführt.

4.2 Kollationierer

Für den Kollationierer, der in TextGridLab 2.0 neu über den Marketplace eingebunden werden kann, wurde die neueste Version der im Rahmen der ESF-COST-Action Interedition[6] entwickelten Kollationierungsbibliothek *CollateX* in das Lab integriert. Dazu wurde ein User Interface entwickelt, das primär auf das Zusammenstellen, Konfigurieren und Anpassen eines Kollationierer-Laufs abzielt.

Dazu wurde ein Verfahren und ein User Interface entwickelt, um Normalisierungen im Kollationierungslauf zu berücksichtigen – als gleich zu behandelnde Tokens können einfach per Drag & Drop konfiguriert werden.

Visualisierungen der Ausgabe wurden zunächst nur in exemplarischer Form umgesetzt. Mittelfristig geplant ist, eine Einbindung der z.Z. in Entwicklung befindlichen Webversion von Juxta[7] zu diesem Zwecke zu prüfen.

[6] http://www.interedition.eu/
[7] http://www.juxtasoftware.org/

5 AP 6.3

5.1 Integration COSMAS

Das COSMAS-Tool erlaubt es Benutzern aus dem TextGridLab in den IDS-Korpera der geschrieben Sprache zu recherchieren. Als Ergebnis wird eine KWIC-Liste (KeyWord In Context) mit Quellenangabe angezeigt. Die KWIC-Liste ist immer eine zufällige Stichprobe aus den Textdaten des Korpus mit maximal 50 Treffern und dient dazu Belege für die Verwendung des Worts zu finden.

Anfragen in COSMAS-II sind direkt aus dem Text oder über einen separaten View möglich. Das COSMAS-Tool steht in TextGridLab 2.0 über das Linguistik-Paket aus dem Marketplace zur Verfügung.

5.2 Integration LEXUS

Die LEXUS-Integration bietet Zugriff auf Lexika des Instituts für Deutsche Sprache und des Max-Planck-Instituts für Psycholinguistik aus dem TextGridLab. Analog zu COSMAS ist die Suche über ein eigenes Suchformular oder direkt aus im Lab bearbeiteten Texten möglich. In TextGridLab 2.0 steht das Tool über das Linguistik-Paket aus dem Marketplace zur Verfügung.

5.3 Integration ANNEX

ANNEX ist ein Tool für die Wiedergabe von Videos gesprochener Sprache synchron zu diesen zugeordneten Annotationen. Eine Integration in TextGridLab 2.0 steht über das Linguistik-Paket aus dem Marketplace zur Verfügung.

6 AP 6.4

6.1 Integration Digilib

Digilib ist ein Tool zur Darstellung von großen Bildern: Es bietet die Möglichkeit, Ausschnitte und Verkleinerungen auch sehr großer Originalbilder zu generieren. Die Integration in TextGrid erfolgte auf Server- wie auf Clientseite.

Der Server konnte so umgebaut werden, dass er auf Streams statt ausschließlich auf lokalen Dateien arbeitet, und damit konnte er an TG-crud und damit an das TextGrid-Repository angebunden werden.

Für die Integration in das TextGridLab wurde ein Plugin entwickelt, das den Server anspricht und entsprechend Bedienelemente für die diversen Funktionen bietet. Im letzten Jahr wurde dieses Plugin wesentlich weiterentwickelt und in seiner Bedienlogik an die des restlichen TextGridLab angeglichen.

Die Digilib-Integration steht mit TextGridLab 2.0 über den Marketplace zur Verfügung.

7 AP 6.5

7.1 Glossen-Editor

Um TextGrid als Editionsumgebung für glossierte, vor allem mittelalterliche, Handschriften zu nutzen, werden verschiedene Komponenten nachgenutzt und auf den spezifischen Glossenfall angepasst: Über den Text-Bild-Linkeditor wird die Beziehung zwischen Digitalisat der Handschriften und Transkription erzeugt, die Glossierung wird im XML-Editor mithilfe einer spezifischen Auszeichnungssprache (Gloss Commentary Markup Language, GCML) erzeugt, die über ein Schema entsprechend angebunden wurde.

Der Aggregationseditor wurde so angepasst, dass er auch zur Erstellung einer speziellen Steuerdatei für den Glossenpublikationsprozess genutzt werden kann, die dann die verschiedenen Elemente einer Glossenpublikation (Digitalisate, Handschriften, Links, Glossierungen) zusammenfasst. Ein Glossen-Publisher erzeugt daraus dann eine Publikation basierend auf einer erweiterten Version des im Rahmen des LMU-Exzellenz-Projektes „Editing Glosses“ erstelltes Präsentationstools (TBL) für Glossen.

Die TextGridLab-Anpassungen für Glossen können in einer Betaversion über den Marketplace in das TextGridLab installiert werden.

8 AP 6.6

8.1 Noteneditor

Mit dem Noteneditor (MEISE) können im TextGridLab MEI-codierte Notentexte dargestellt und bearbeitet werden. Die Ansicht ist sowohl in grafischer Form analog zu einem Notenblatt als auch in Form eines detaillierten Strukturbaums möglich. Der Fokus liegt dabei allerdings weniger auf einer mit expliziten Notensatzwerkzeugen vergleichbaren grafischen Darstellung als vielmehr auf besonderen Fähigkeiten im editorischen Bereich wie die Darstellung von Varianten.

Im letzten Jahr wurden das Tool in seiner Funktionalität wesentlich erweitert – unter anderem wurde die Komponente zur grafischen Notenansicht komplett überarbeitet – und steht mit Version 2.0 nun über den Marketplace zur Integration ins TextGridLab bereit.

9 AP 6.8

9.1 OCR

Es wurde ein OCR-Service implementiert, der Input-Dokumente im TIFF-, PNG- und JPG-Format nach Prozessierung im XHTML+hOCR-Format ausgibt. Es wurde eine Umgebung zum interaktiven Clustering und Labeling von Buchstaben und Ligaturen entwickelt, die nun für die Buchstabenerstellung in Fraktur genutzt werden kann., und damit ein Fraktur-Trained-Model konnte mit den Trainingsdaten von Theodor Fontanes „Wanderungen durch die Mark Brandenburg“ erarbeitet.

Zudem wurde OCRopus erweitert, sodass u.A. Unicode-Unterstützung und Ligaturen, große Zeichensätze bei den Klassifikatoren und Unterstützung für sehr große Datenmengen beim Training möglich waren, dies ist für die Frakturunterstützung nötig.

Der OCR-Service kann über das Workflowtool in TextGridLab 2.0 eingebunden werden.