TextGrid
Virtuelle Forschungsumgebung
für die Geisteswissenschaften

Dokumentation und Leitfaden für den TextGrid-Import (R 4.4.1)

Version 19. September 2013
Arbeitspaket 4.4
verantwortlicher Partner SUB Göttingen

TextGrid
Virtuelle Forschungsumgebung für die Geisteswissenschaften

Ein Projekt im D-GRID

GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

Projekt: TextGrid – Institutionalisierung einer Virtuellen Forschungsumgebung in den Geisteswissenschaften

BMBF Förderkennzeichen: 01UG1203A

Laufzeit: Juni 2012 bis Mai 2015

Dokumentstatus: final

Verfügbarkeit: öffentlich

Autoren:

Maximilian Brodhun, SUB Göttingen

Stefan E. Funk, SUB Göttingen

Roman Hausner, SUB Göttingen

Mathias Göbel, SUB Göttingen

Ubbo Veentjer, SUB Göttingen

Revisionsverlauf:

Datum	**Autor**	**Kommentare**
1. Juli 2013	Stefan E. Funk	Erste Version und Themensammlung
17.Juli 2013	Maximilian Brodhun	Strukturierung und erste Texte
23. Juli 2013	Maximilian Brodhun	Weitere Texte und Übersetzungen ins Deutsche
24. Juli 2013	Maximilian Brodhun	Weitere Texte, leichte Anpassung der Strukturierung
30. Juli 2013	Maximilian Brodhun	Angepasste Strukturierung, Formatierung, Korrektur der Texte
31. Juli 2013	Maximilian Brodhun	Weitere Korrekturen, Screenshots eingefügt, Abbildungs- und Tabellenverzeichnis
21. August 2013	Mathias Göbel	Kapitel 6.1. eingefügt
22. August 2013	Maximilian Brodhun	Kleinere Korrekturen
22. August 2013	Mathias Göbel	Kleinere Korrekturen
2. September 2013	Maximilian Brodhun	Anpassung: einheitliches Vokabular
3. September 2013	Stefan E. Funk	Strukturierung und allgemeine Überarbeitung
4. September 2013	Stefan E. Funk	Mehr überarbeitet und Ubbos Blumenbach-Part eingefügt
13. September 2013	Stefan E. Funk	Endredaktion
19. September 2013	Maximilian Brodhun	Kommentare eingearbeitet

Inhalt

1. Virtuelle Forschungsumgebung für die Geisteswissenschaften

Das hauptsächliche Ziel von TextGrid ist die Bereitstellung einer virtuellen Forschungsumgebung[1] für die Geisteswissenschaften, mit der es durch die Nutzung verschiedener Anwendungen und Services möglich ist, Texte und Bilder zu erstellen, zu editieren und zu publizieren. TextGrid besteht aus den beiden Hauptkomponenten *TextGrid Laboratory* (TextGridLab) und *TextGrid Repository* (TextGridRep).

Die Anwendungen und Services des TextGridLab sind zunächst auf die spezifischen Bedürfnisse der Geisteswissenschaften wie Philologie, Linguistik, Musikwissenschaften und Geschichte ausgerichtet. Des Weiteren unterstützt TextGrid die Speicherung und Nachnutzung von Forschungsdaten durch die Integration in das TextGridRep, in welchem Daten aufbewahrt und zugreifbar gemacht werden. Die Architektur von TextGrid ist erweiterbar, zusätzliche Anwendungen und Services können zu den bestehenden hinzugefügt werden.

Das grundlegende Konzept von TextGrid ist, dass Geisteswissenschaftlerinnen und Geisteswissenschaftler nicht mehr durch die Grenzen der individuellen Arbeitsplätze beschränkt sein sollen und kollaborativ in Projekten zusammenarbeiten können, ohne dabei abhängig von der geographischen Position des Arbeitsplatzes zu sein. Dabei soll weitergehend garantiert werden, dass die Daten sicher und zuverlässig gespeichert werden (entsprechend den Richtlinien der Guten Wissenschaftlichen Praxis [2]).

Das *TextGridLab* ist eine plattformunabhängige Open-Source Software. Sie ermöglicht einen integrierten Zugang zu speziellen Anwendungen, Services und Inhalten. Eine Vielzahl von erprobten Anwendungen, Services und Ressourcen, die den gesamten Arbeitsprozess unterstützen (z.B. die Erstellung von textuellen Editionen) werden dabei geboten.

Das *TextGridRep* ist ein Forschungsdatenarchiv, welches die langfristige Speicherung und Nachnutzung von Forschungsdaten ermöglicht. Sind Dokumente im TextGridRep einmal publiziert, können sie nicht wieder gelöscht werden. Updates und neue Editionen eines TextGrid Objektes können hinzugefügt werden (Revisionierung). Jedoch bleibt das ursprüngliche Objekt immer bestehen, um den Richtlinien der Guten Wissenschaftlichen Praxix zu genügen. Der Persistent-Identifier-Service der GWDG[3] erstellt einen persistenten Identifikator (PID, in dem Fall einen Handle[4]) für jedes in TextGrid publizierte Objekt. Durch diese PID ist eine eindeutige und langfristifge Referenzierung garantiert.

Im Gesamten betrachtet spricht TextGrid drei Hauptnutzergruppen an:

- Geisteswissenschaftlerinnen und Geisteswissenschaftler, die an Forschungsprojekten und digitalen Editionen arbeiten,

1 Virtual Research Environment (VRE)
2 http://www.dfg.de/foerderung/grundlagen_rahmenbedingungen/gwp/
3 http://www.pidconsortium.eu/index.php?page=home
4 http://handle.gwdg.de:8080/pidservice/

- Softwareentwicklerinnen und Softwareentwickler, die neue Anwendungen und Services für TextGrid implementieren möchten, und

- „Content Providers“ (z.B. Archive und Forschungsinstitute), die ihre Daten in TextGrid bereitstellen und einem größeren Nutzerkreis zur Verfügung stellen möchten.

2. Einführung in den TextGrid Datenimport

In der zweiten Förderphase des Projektes TextGrid[5] wurden bereits umfangreiche Datensammlungen in das TextGrid Repository eingespielt, wie z.B. die Digitale Bibliothek[6]. Um weiteren – auch umfangreichen – Datensammlungen eine Archivierung im TextGrid Repository zu ermöglichen, wird die vorhandene technische Dokumentation des externen Import-Tools (hier wird die Java-Bibliothek koLibRI genutzt) unter Einbeziehung importierender Nutzerinnen und Nutzer weiter verfeinert und ergänzt.

Darüber hinaus wird eine inhaltliche Dokumentation erstellt, die für einen Import erforderliche Kenntnisse vermittelt, wie etwa das von TextGrid für den Import vorgegebene Metadatenschema, Metadaten-Konversion oder -Mapping etc. Fehler im Import-Modul der Java-Bibliothek koLibRI werden, falls erforderlich, ebenfalls behoben. Weiterhin wird die SUB Göttingen bei der Einspielung ihrer Datensammlungen technischen und inhaltlichen Support sowie Beratung bieten. Eine Beratung z.B. bei der Strukturierung der einzuspielenden Daten sowie deren verfügbarer Metadaten ist in diesem Zusammenhang von eminenter Bedeutung, ebenso wie auch die Vorbereitung auf die nachträgliche Bearbeitung der Daten im TextGridLab[7] und die anschließende Publikation per TG-publish bzw. die direkte Publikation im TextGridRep[8].

Für die Präsentation der eingespielten Daten kann zunächst der existierende TextGridRep-Browser genutzt werden, der Zugriff auf alle im TextGridRep vorhandenen Daten ermöglicht. Für die Präsentation projektspezifischer Daten kann jederzeit ein im Aussehen sowie im Datenbestand angepasster Repository-Browser aufgesetzt werden. Darüber hinaus wird eine Referenzinstanz eines Präsentations-Frameworks entwickelt, das eine feingliedrige Darstellung projektspezifischer Inhalte erlaubt. Hierzu kann möglicherweise das Framework SADE genutzt werden, beispielsweise in Verbindung mit einer projekteigenen eXist-Installation, die aus dem TextGrid Repository die benötigten Daten kopiert und entsprechend den Projektanforderungen visualisiert.

5 http://www.textgrid.de/
6 http://www.deutsche-digitale-bibliothek.de/
7 http://www.textgrid.de/registrierungdownload/download-und-installation/
8 http://www.textgridrep.de/

3. TextGrid-Datenimport

Die Import-Funktion von TextGrid wird genutzt, um Dateien zu speichern und zu öffnen, die nicht durch das TextGridLab erzeugt wurden. Dies gilt auch für den Fall, wenn Objekte zwar im TextGridRep gespeichert sind, aber (noch) nicht in das TextGridLab importiert wurden.

In einigen Fällen wurden Objekte vorher aus dem Lab gelöscht, sodass sie bei Bedarf re-importiert werden müssen, um an diesen Objekten Editierungsarbeiten durchführen zu können.

Gewöhnlich werden Dateien durch ihren Namen und den Pfad identifiziert, unter dem sie gespeichert sind. Um in TextGrid ein Objekt eindeutig identifizieren zu können, wird beim Import eine URI (Uniform Resource Identifier) vergeben, die sogenannte TextGrid URI.

Im Folgenden werden die zwei grundsätzlichen Arten beschrieben, um Dateien in TextGrid zu importieren.

3.1 Beschreibung notwendiger Software

Für den Datenimport in TextGrid sind drei verschiedene Anwendungen relevant. Diese sind die beiden Hauptkomponenten von TextGrid, das TextGridLab und das TextGridRep, sowie die externe Anwendung koLibRI. Diese werden im Folgenden in ihren Grundzügen beschrieben.

3.1.1 koLibRI

Die koLibRI (kopal Library for Retrieval und Ingest) ist eine Bibliothek für die Abbildung von Workflows, die durch modulisierbare Programmteile zum Datenimport in TextGrid mit Workflow- und Hotfolder-Mechanismen genutzt werden kann.

Die koLibRI-Anwendung wurde im kopal-Projekt[9] zum Erstellen, Einspielen und Abrufen von Langzeitarchivierungs-Daten (SIPs und DIPs)[10] entwickelt und von TextGrid für die Metadatenprozessierung und das Einspielen von Daten in das TextGridRep erweitert. Außerdem wird koLibRI als Grundlage für die Dienste TG-publish und TG-copy genutzt, die für die Publikation von Daten aus dem TextGridLab und das Kopieren innerhalb desselben verantwortlich sind. Eine Anleitung für die Nutzung dieser Dienste ist über das öffentliche TextGrid-Wiki[11] abrufbar.

3.1.2 TextGridLab und TextGridRep

Die beiden Komponenten TextGridLab und TextGridRep wurden in Kooperation mit Geisteswissenschaftlerinnen und Geisteswissenschaftlern entwickelt und zielen auf die Bedürfnisse von Wissenschaftlern, die die Forschungsmöglichkeiten mittels digitaler

9 http://kopal.langzeitarchivierung.de/index_koLibRI.php.de
10 SIP – Submission Information Package, DIP – Dissemination Information Package
11 https://dev2.dariah.eu/wiki/display/TextGrid/TG-publish

Technologien entdecken möchten. Dies gilt sowohl für individuelle Forscher, als auch innerhalb einer Community von geographisch voneinander getrennten Forschern.

Eine stetig steigende Anzahl von Geisteswissenschaften arbeiten mit komplexen Forschungsfragen und großen Mengen an Daten. In diesem Zusammenhang ist häufig ein interdisziplinäres Forscherumfeld tätig. Die kollaborative Arbeit in diesem Umfeld wird durch eine Anwendung wie TextGrid deutlich vereinfacht, in der sowohl Projekt- als auch Nutzeradministration integrierte Komponenten sind.

Durch die Nutzung von TextGrid können Wissenschaftler produktiv in einer sicheren digitalen Umgebung zusammenarbeiten. Eine zuverlässige Speicherung der Forschungsdaten wird innerhalb dieser Umgebung garantiert. Die in TextGrid verfügbaren Anwendungen und Dienste sind flexibel, erweiterbar und anpassbar an verschiedene Forschungsmethoden. Zusätzlich werden diese Anwendungen aufrechterhalten und fortlaufend aktualisiert, um einen langfristigen Zugriff und eine Nachnutzung der Forschungsdaten zu sichern.

Eine virtuelle Forschungsumgebung wie TextGrid ermöglicht neue Formen der Kollaboration und Forschung an einem sicheren und zuverlässigen Arbeitsplatz. Die folgenden Features werden durch TextGrid den Geisteswissenschaften zur Verfügung gestellt.

Administration und Workflow-Organisation

Dieses Feature besteht aus Services, die es ermöglichen, Projekte zu administrieren und zu organisieren:

- Einem fortgeschrittenem User-Management, welches die Vergabe und die Administration von Rollen überwacht und die Zugriffsrechte verwaltet,
- einem effizienten Dokumenten-Managementsystem, welches effektiv Projekte und Objekte auf verschiedenen Ebenen verwalten kann und dabei komplexe Beziehungen zwischen den Objekten beachtet,
- einem Workflow-Tool, welches eine semi-automatische Prozessierung der Daten ermöglicht, sowie
- einem Versions-Management, um sicherzustellen, dass der Arbeitsprozess durch verschiedene Revisionen umfassend dokumentiert werden kann. Dokumente, die endgültig veröffentlicht wurden, können nicht gelöscht werden, jedoch besteht die Möglichkeit, neue Versionen und Updates eines Objektes dem Repository hinzuzufügen.

Dezentraler Arbeitsplatz

Nutzerinnen und Nutzer können TextGridLab und TextGridRep unabhängig vom geographischen Ort benutzen und gemeinsam an komplexen Forschungsprojekten arbeiten.

Standardisierung

Das von TextGrid festgelegte Metadaten-Vokabular und die von TextGrid unterstützten offenen Standards erleichtern den Fachwissenschaftlern das Austauschen und Abfragen von Daten und Texten und bieten Möglichkeiten für die digitale Archivierung.

Erweiterungsmöglichkeiten

TextGrid ist erweiterbar und ermöglicht es damit, bestehende Anwendungen zu verbessern oder neue Tools und Services hinzuzufügen; entweder als externe Web-Services oder als interaktive Services, welche die Eclipse Rich Client Plattform benutzten, auf der das TextGridLab basiert.

Modularität

Das TextGridLab ist modular angelegt. Solche Module sind zum Beispiel der XML-Edtitor, welcher als eines der Kern-Elemente des TextGridLab agiert und auf dem die VRE aufbaut. Eine große Anzahl von Tools und Services sind mit dem XML-Editor assoziiert und können arrangiert und miteinander kombiniert werden, je nach dem, wie es die spezifischen Bedingungen der Nutzerinnen und Nutzer und des Projektes erfordern. Einige Tools erlauben das Einfügen von Elementen in den Quelltext des XML-Editors, während andere eine strukturelle Verbindung zum XML-Editor besitzen. Die offenen Schnittstellen ermöglichen die Integration von neuen und speziell entwickelten Anwendungen. Diese modulare Struktur erleichtert den Arbeitsprozess innerhalb der virtuellen Forschungsumgebung und deren Erweiterung.

Nutzerverwaltung

TextGrid bietet ein *rollenbasiertes Zugriffskontrollsystem*, das mit OpenRBAC[12] implementiert ist. Jedes TextGrid-Projekt hat eine Nutzerrolle, die für die Projektmanagementaufgaben verantwortlich ist, der sogenannte *Projekt-Manager*. Diese Rolle hat als einzige die Möglichkeit, Rollen an andere Nutzer zu delegieren. Nur Mitglieder eines Projektes haben Zugriff zum – noch nicht publizierten – Inhalt des jeweiligen Projekts. Bei der Nutzung dieses Systems ist es möglich, dass temporär bestimmte Nutzerrollen vergeben werden. Dies kann der Fall sein, wenn zum Beispiel eine kurze Teilnahme bestimmter Personen notwendig ist, um spezifische Aufgaben innerhalb des Projektes zu übernehmen. Somit ist es für den Projekt-Manager möglich, die Zugriffsrechte auf die Projektdaten zu kontrollieren. Durch dieses System kann die sichere und zuverlässige Zusammenarbeit garantiert werden, sowie die gespeicherten Daten innerhalb der Forschungsumgebung weltweit erreichbar sein. Weitere Rollen sind *Administrator*, *Bearbeiter* und *Beobachter*.

Verteilter Speicher

Für die Suchfunktion im Repository von TextGrid wurde zusätzlich zum hauptsächlichen Datenspeicher eine verteilte Speicherstrategie erstellt. Dieser verteilte Datenspeicher besteht aus zwei Teilen, dem *Suchindex 1* und dem *Suchindex 2*. Die initiale Aktivierung, Erstellung und Bearbeitung von Dateien wird im TextGridRep gespeichert. Die Daten werden im Suchindex 1 gespeichert und haben hier noch keine PID. Dieser erste Suchindex ist dynamisch, das heißt, dass die Daten nur über das TextGridLab zugreifbar sind und nur von Nutzerinnen und Nutzern, die eine Rolle innerhalb des Projektes zugesprochen bekommen haben. Diese Daten können nur verändert oder gelöscht werden, wenn die spezifische Rolle die entsprechenden Rechte einräumt.

12 http://www.openrbac.de/en_startup.xml

Nachdem durch die Nutzerrolle *Projekt-Manager* entschieden wurde, die Daten im endgültigen Format zu publizieren, werden sie durch den Publikations-Prozess aus dem TextGridLab unter Anderem in den zweiten Suchindex kopiert (Suchindex 2). Während dieses Prozesses erhalten alle Objekte unter Anderem auch eine PID, um eindeutig referenzierbar zu sein und dies auch zu bleiben. Die transferierten und publizierten Dateien können nicht mehr gelöscht oder verändert werden. Spätere Modifikationen am Datenbestand können nur durch Kopieren und einen anschließenden Re-Import in Suchindex 1 erreicht werden.

Suchindex 2 ist öffentlich zugreifbar. Da die Daten unveränderlich sind, ist dieser Suchindex statisch. Große Datenmengen können auch aus anderen Archiven oder Publikations-Servern hochgeladen werden, um sie durch das TextGridRep erreichbar zu machen. Sowohl TextGridLab Nutzerinnen und Nutzer als auch öffentliche Nutzerinnen und Nutzer können den Suchindex 2 durchsuchen. Im Zusammenspiel mit den vergebenen PIDs ist auch die Integration des TextGridRep in andere Repositorien denkbar.

4. Objekte im Allgemeinen und TextGrid-Objekte im Speziellen

TextGrid verwaltet Objekte und die verschiedenen Beziehungstypen zwischen den Objekten. Diese Beziehungen sind durch Aggregationen, Editionen und Kollektionen dargestellt. Um Beziehungen zwischen Objekten zu erstellen, ist jedes Objekt eindeutig identifizierbar. Um eine effiziente Identifikation (vor Vergabe der URIs) der Daten und eine Suche über verschiedene Projekte zu ermöglichen, muss ein jedes Objekt durch Metadaten beschrieben werden. Im Folgenden werden die verschiedenen Objektarten beschrieben.

TextGrid-Objekte bilden die Basisdateneinheit in TextGrid. Alles was in TextGrid gespeichert wird (XML-Dokumente, Bilder, usw...), wird als TextGrid-Objekt behandelt und auch gespeichert.

Ein TextGrid-Objekt besteht aus einem Metadatensatz und einer Inhaltsdatei (z.B. Volltextdaten oder Bilder). Beides wird vom TextGridLab und den Middleware-Services im Allgemeinen zusammen verarbeitet:

Wird der Inhalt eines Objektes in einem Editor bearbeitet, besteht die Möglichkeit, in den Metadateneditor des TextGridLab zu wechseln und den korrespondierenden Metadatensatz zu ändern. Dieser wird in dem Moment gespeichert, in dem auch das gesamte Objekt gespeichert wird, oder auf Wunsch auch einzeln. Bei der Nutzung des Metadateneditors sind hauptsächlich deskriptive und bibliographische Metadaten sichtbar und editierbar. Der komplette Metadatensatz zu einem Objekt wird durch die Nutzung der Funktion *Show functional metadata* im Kontextmenüs des Objektes als originäre XML-Struktur sichtbar. Der Metadatensatz kann durch Nutzung des Template-Editors selbstständig erstellt werden. Um diesen nutzen zu können ist die Rolle *Projekt-Manager* notwendig.

Jedes Objekt wird durch eine URI identifiziert (zum Beispiel: *textgrid:74k0.1)*. Diese sogenannte TextGrid-URI wird automatisch generiert, wenn ein Objekt in das TextGrid Repository eingespielt wird. Diese URI kann durch die Nutzung von *Copy URI*, im Kontextmenü des Objektes, in die Zwischenablage kopiert werden. Ebenfalls kann die URI in der Ansicht der technischen Metadaten gefunden werden. Zusätzlich ist die URI eines Objektes in der Statusbar des Objektes sichtbar, wenn es im Navigator ausgewählt wurde. Ein Objekt immer zu einem Projekt zugehörig. Dabei besteht weitergehend die Option Objekte zu Aggregationen zuzuordnen, die in Relation zu einem anderen Projekt stehen. Somit können Kreuzreferenzen entstehen.

Alle Objekte haben einen Content-Type. Diese Information ist eine Pflichtangabe im Metadatenschema. Ein Beispiel für diesen Content-Type ist *text/xml* oder *image/jpeg*, es existieren aber auch TextGrid-spezifische Typen wie z.B. *text/tg.aggregation+xml*.

Projekte sind Container für das Rechtemanagement der Nutzer. Des Weiteren ist jedes TextGrid Objekt vom Typ *Item (Element)*, *Edition (Edition)*, Werk (*Work)* oder Kollektion (*Collection)*. Diese Objekte haben spezifische Metadaten und Semantiken.

4.1 Element

Elemente sind die einfachste und gleichzeitig häufigste Entität in TextGrid. Ein Item kann zum Beispiel eine eingescannte Buchseite im XML-Format sein. Eine Auswahl an Items kann durch Klick auf ein Symbol in der Toolbar erzeugt werden. Solche Items können zum Beispiel XML-Dokumente, Klartextdokumente, Stylesheets oder Musiknotationen sein. Items sind abhängige Objekte: Sie können nur als Teil einer Edition oder Kollektion publiziert werden. Die deskriptiven Metadaten eines Items bestehen generell – für die Publikation verpflichtend – aus einem (oder mehreren) Titeln, dem Format, dem Rechteinhaber und den identifizierbaren technischen Metadaten (diese werden automatisiert erzeugt). Für ein erstes Anlegen oder Importieren eines Items sind lediglich ein Titel und das Format verpflichtend. Um weitere Metadaten, wie einen Autor oder eine Quellenbeschreibung, anzugeben, muss das Item Teil einer Edition sein und diese kann optional mit einem Werk assoziiert werden.

4.2 Aggregationen

Eine *Aggregation* ist ein TextGrid-Objekt, das aus einer geordneten Liste von Referenzen auf andere Objekte besteht, welche ebenfalls selbst Aggregationen sein können.

Aggregationen werden genutzt, um TextGrid-Objekte zu organisieren. Sie können ähnlich zu den Ordnern eines Betriebssystems genutzt werden. Allerdings kann ein Objekt auch mehreren Aggregationen zugeordnet werden und somit in einer Aggregation gesammelt werden und Nutzern gehören und welche eventuell nur gelesen werden können. Wenn eine Aggregation gelöscht wird, werden die aggregierten Objekte gewöhnlich aufrecht erhalten, außer es wird der explizite Befehl zum Löschen aller aggregierten Objekte gegeben.

Innerhalb einer Aggregation können verschiedene Objekte vorhanden sein, die sich durch ihre Semantiken und Metadaten unterscheiden. Gemeinsam haben allerdings alle, dass sie eine Art von Kontext haben. Einfache Aggregationen besitzen lediglich nur ein weiteres Objekt, und können benutzt werden, um beispielsweise ein einzelnes Buchkapitel zu repräsentieren, welches über mehrere Elemente verteilt ist. Editionen oder Kollektionen sind technisch gesehen ebenfalls Aggregationen, definiert durch einen charakteristischen Metadatensatz. Der Aggregationseditor kann genutzt werden, um Objekte in einem verschachtelten Datensatz von Aggregationen zu erstellen und zu verwalten.

4.3 Editionen

Eine Edition ist die Manifestation eines Werkes. Sie ist eine spezielle Form einer Aggregation und enthält weitere benötigte Metadatenfelder, um die Edition näher zu beschreiben. Dies sind beispielsweise die Personen und Organisationen, die an der Vorbereitung der Edition beteiligt sind, oder auch die Quelle (außerhalb von TextGrid; beispielsweise ein Buch), welches die elektronische Edition repräsentiert. Eine Edition ist häufig mit einem Werk, in der prä-publizierten Form, assoziiert und muss mit diesem Werk für die Publikation assoziiert sein. Editionen können publiziert werden, wenn sie die verpflichtenden Metadaten assoziiert mit dem entsprechenden Werk besitzen und mindestens ein Item besitzen.

4.4 Kollektionen

Die Kollektion ist eine Akkumulation von TextGrid-Objekten. Kollektionen sind für zwei Nutzungsfälle konzipiert: In einem organisatorischen Kontext kann eine Edition Teil einer Kollektion sein (z.B. die Kombination von Editionen des Blumenbachprojekts unter dem Label „Blumenbach“ oder die Kombination von digitialisierten Drucken des 18. Jahrhunderts unter dem Label VD 18). Zusätzlich können Kollektionen genutzt werden, um Items zu aggregieren, für die das Edition/Werk-Konzept nicht anwendbar ist (z.B. die Digitalisierung von Museumsstücken).

4.5 Werk

Ein Werk ist eine individuelle Kreation, z.B. ein literarisches Opus, welches in verschiedenen Editionen verfügbar sein kann (z.B. als Taschenbuchausgabe, als Teil einer Serie von gesammelten Werken, als eine theatralische Performance oder als Hörbuch). Eine Edition ist immer Teil eines bestimmten Werk. Um ein Werk zu beschreiben, bietet TextGrid Werk-Objekte an, welche ausschließlich aus Metadaten bestehen und die spezifische Metadaten wie den Titel, das Erstellungsdatum und Schlüsselwörter beinhalten. Beim Publikationszeitpunkt muss jedes Editions-Objekt in Relation zu einem Werk stehen.

4.6 Referenzen zwischen den Objekten

Es besteht die Möglichkeit, Referenzen bzw. Beziehungen zwischen verschiedenen Arten von Objekten auf zwei Weisen zu erstellen:

1. Um eine Edition in Relation zu einem Werk hinzuzufügen, kann der Metadateneditor des TextGridLab genutzt werden. Die Eingabemaske des Metadateneditors kann durch Auswahl der Edition in der Navigation geöffnet werden. Mit einem Rechtsklick und durch die Auswahl *Metadaten öffnen*, werden diese Metadaten angezeigt und sind daraufhin editierbar. Im daraufhin sich öffnenden Metadateneditor muss das Feld *Edition of* ausgewählt und danach der Button *Browse...* betätigt werden. Relevante Werke für diesen Vorgang werden angezeigt. Wird ein Werk ausgewählt, wird dessen TextGrid URI im Feld *Input* angezeigt. Damit ist die Beziehung zwischen dem Werk und der Edition hergestellt.

2. Die Erstellung von Referenzen zwischen Items und Editionen oder Kollektionen ist durch den Aggregations-Editor möglich. Um diesen zu öffnen, muss ein Rechtsklick auf die Edition oder Kollektion im Navigator vorgenommen werden. Durch die Auswahl *Bearbeiten* im Kontextmenü öffnet den Aggregations-Editor. Daraufhin können ein oder mehrere Items vom Navigator in die entsprechende Aggregation, Edition oder Kollektion gezogen werden. Im Navigationsbaum sind die Zusammengehörigkeiten zwischen Edition und Item erkennbar. Diese Ansicht ist auch im Metadatenfeld *Part of Edition(s)* vorhanden. Dafür müssen die Metadaten zu einem Item geöffnet werden. Dort ist ebenfalls erkennbar, ob ein Item in Relation zu einer Edition steht.

5. Grundsätzliche Arten des Imports in das TextGrid Repository

Generell existieren zwei verschiedene Arten des Imports in das Repository von TextGrid. Zum einen kann die Importfunktion des TextGridLabs genutzt werden und zum anderen ist es möglich, den Import mittels des Moduls TG-Import durchzuführen, welcher auf koLibRI aufbaut. In diesem Abschnitt werden diese beiden Arten in den Grundzügen beschrieben, um später eine detaillierte Anleitung zum Import geben zu können.

5.1 Das TextGridLab Import-Modul

Das im TextGridLab bereit gestellte Import-Modul ermöglicht den Import lokaler Dateien und auch ganzer lokaler Verzeichnisstrukturen in den dynamischen Bereich des TextGrid Repositories. Der dynamische Bereich ist derjenige, in dem Dokumente abgelegt werden, die vom TextGrid-Rechtemanagement verwaltet werden und somit nur für angemeldete und freigeschaltete Personen sichtbar und ggf. auch editierbar sind.

Dieser Speicherbereich wird für eine kontinuierliche und kollektive Arbeit an nicht öffentlichen Dokumenten benutzt. Die über das Import-Modul des TextGridLab eingespielten Dokumente werden alle über den Speicherdienst TG-crud in diesen dynamischen Speicherbereich geschrieben, sowie auch in den Suchindex , für den die XML-Datenbank eXist und die RDF-Datenbank Sesame eingesetzt werden. Schließlich werden alle Dokumente beim Rechtemanagement registriert und sind so vor unerlaubtem Zugriff geschützt. Beim Import von Daten in das TextGridLab werden in den Dokumenten enthaltene Referenzen auf andere zu importierende Dokumente von lokalen Dateipfaden nach (während des Einspielvorgangs automatisch generierten) TextGrid-URIs umgeschrieben, so dass diese Referenzen innerhalb des TextGridRep ihre Gültigkeit nicht verlieren.

5.2 Importieren mit TG-Import

Eine weitere Möglichkeit des Datenimports bietet die Software-Bibliothek TG-Import. . Diese Methode dient zuvorderst dem direkten Import – der Publikation – auch großer Datenmengen in das öffentliche TextGrid Repository, mithin in den statischen Speicherbereich und Suchindex 2.Durch Konfiguration ist es jedoch möglich, diesen Ingest auch für den Datenimport in den dynamischen Speicherbereich des TextGridRep und den Suchindex 1 zu nutzen. So können auch Daten im TextGridLab verfügbar gemacht werden, die beispielsweise individuell vorbereitet werden. TG-Import erweitert die in den Projekten kopal und DP4lib entwickelte Java-Bibliothek koLibRI mit diversen Modulen und Diensten für die Nutzung mit TextGrid.

Ebenso wie im TextGridLab Import-Modul besteht die Möglichkeit, ganze Verzeichnisstrukturen einzuspielen, wobei die Struktur des Quellordners übernommen wird. Für den fortgeschrittenen Nutzer- bzw. Entwicklerkreis mit TextGrid-Objekt-, XML- und XSLT-Kenntnissen kann auch ein Import von fertig vorbereiteten Datenobjekten direkt durchgeführt werden, wobei dann die TextGrid-Metadaten, Referenzen, TextGrid-URIs und

weitere benötigte und gewünschte Datenstrukturen selbst erzeugt werden können bzw. vor dem Importieren vorliegen müssen; dies ermöglicht einen sehr flexiblen Import.

Darüber hinaus ist ein Einspielen von DFG-Viewer METS-Dateien möglich, wobei die dort enthaltenen Metadaten und referenzierten Dokumente (z.B. Scans von Buchseiten und/oder Handschriften) heruntergeladen und in das TextGridRep kopiert werden. In einer METS-Datei können generell Struktur- und Metadaten eines digitalen Objekts strukturiert beschrieben werden, in einer DFG-Viewer METS-Datei sind diese Strukturen nach einer bestimmten Spezifikation abgelegt – dem zvdd/DFG-Viewer METS-Profil[13]. Dies beinhaltet eine festgelegte Struktur sowie bestimmte Metadaten und auch Referenzen auf die Digitalisate selbst. So können die Digitalisate mitsamt ihrer Struktur und deren dazugehörigen Metadaten mittels des DFG-Viewers anschaulich dargestellt werden. Beim Import der Daten in das TextGridRep werden die Referenzen auf die Bilder in der METS-Datei so umgeschrieben, dass sie auf die importierten Digitalisate im TextGridRep verweisen. Außerdem werden die in der METS-Datei enthaltene logische und physikalische Struktur sowie alle gewünschten Metadaten, konfigurierbar über anpassbare XSLT Stylesheets, übernommen.

13 DFG-Viewer – Profil der Metadaten. http://dfg-viewer.de/profil-der-metadaten/

6. Anleitungen für die Importvorgänge

In diesem Abschnitt sollen die Möglichkeiten des Datenimports in TextGrid im Detail beschrieben werden. Dabei wird auf die Möglichkeiten des Imports lokaler Dateien eingegangen, sowie den Re-Import von Daten, die bereits einmal in TextGrid importiert waren. Im Anschluss folgt eine Detailbeschreibung des Imports von externen Dateien mit der Nutzung der Bibliothek koLibRI.

Um das Import-Tool des TexGridLab benutzen zu können, muss unter *Datei* der Eintrag *Lokale Dateien importieren* ausgewählt werden. Daraufhin öffnet sich ein Kontextmenü, welches in Abbildung 1 zu sehen ist.

Das Import-Tool kann sowohl genutzt werden, um neue Dateien in TextGrid zu importieren, als auch vorher exportierte Dateien zu re-importieren. Für beide Punkte wird folgend das Vorgehen beschrieben.

6.1 Importieren von lokalen Dateien

1. Die Import-Ansicht ist über das Hauptmenü aufrufbar Lokale Dateien importieren ... (siehe Abbildung Abbildung 1).

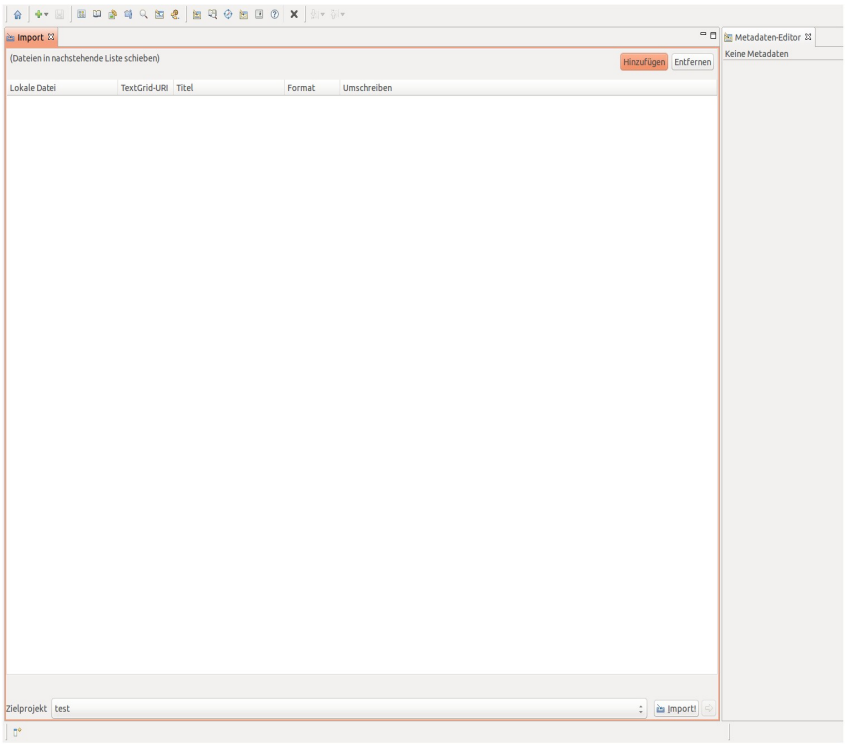

Abbildung 1: Die Import-Ansicht vor Auswahl zu importierender Dateien

2. Es muss ein Projekt ausgewählt werden, in welches die gewünschten Dateien importiert werden. Das Projekt kann durch das Dropdown-Menü in der unteren Bildschirmsektion ausgewählt werden (siehe Abildung Abbildung 1).

3. Im Folgenden können einzelne Dateien oder komplette Ordner durch das Dateimanagementsystem des Betriebssystems in die Liste der zu importierenden Dateien des Import-Tools gezogen werden. Alternativ zu diesem Vorgehen können Dateien auch durch den Button *Hinzufügen* hinzugefügt werden. Um Dateien zu löschen, kann der Button *Löschen* genutzt werden (siehe Abildung Abbildung 2).

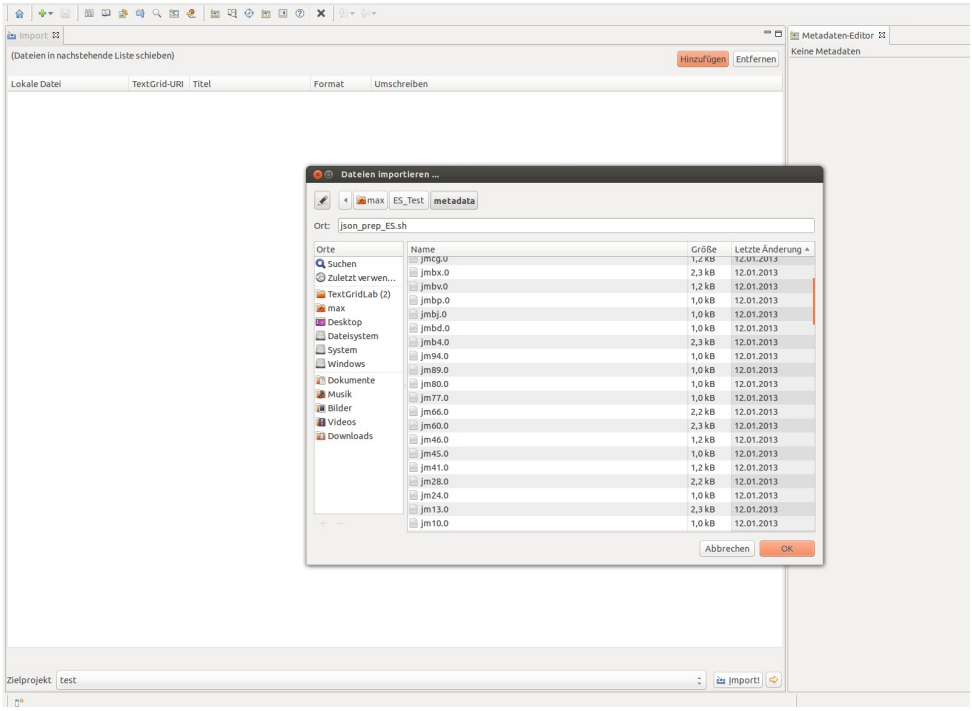

Abbildung 2: Auswahl von Dateien bzw. Ordner durch den Dateimanager des Betriebssystems

Alternativ zu diesem Vorgehen können Dateien auch per Drag & Drop in die Importansicht gezogen werden (siehe Abbildung 3).

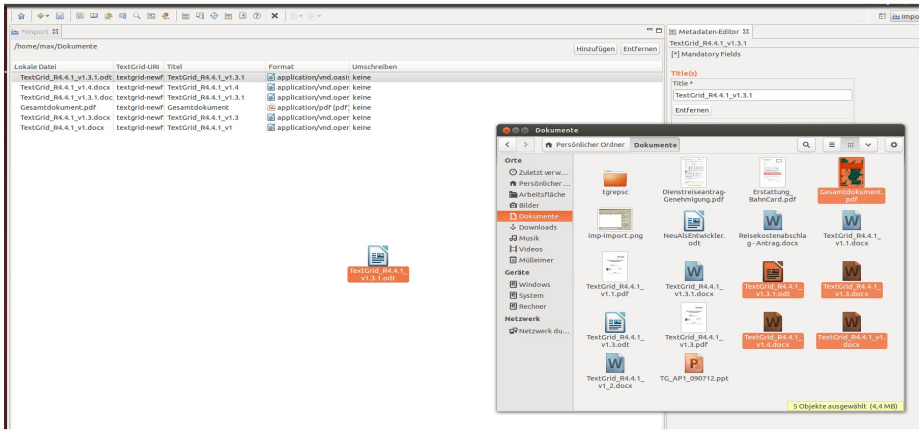

Abbildung 3: Auswählen von zu importierenden Dateien mittels „Drag & Drop“

4. Das Import-Tool gibt Vorgaben für den Format, Titel und eine mögliche Umschreibung der Daten zu in TextGrid gebräuchlichen Auszeichnungssprachen. Diese sind jedoch nur als Vorschlag zu verstehen und können durch einen Klick auf einen bestimmten Eintrag in der Tabelle geändert werden. Bereits vor der Ausführung des Importsvorganges kann der Metadateneditor zur erweiterten Beschreibung und Vervollständigung der Metadaten genutzt werden (siehe Abildung Abbildung 4).

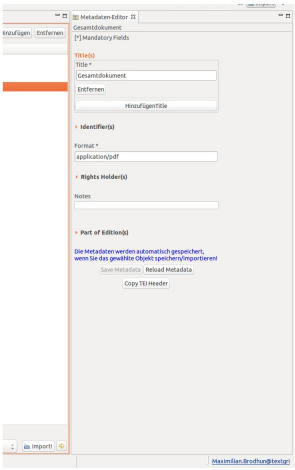

Abbildung 4: Ansicht des Metadateneditors zum Ändern und Eintragen bestimmter Metadatenfelder

5. Nach diesen Vorbereitungsschritten und einen Klick auf den Button *Import* wird der Importvorgang gestartet (siehe Abbildung 5). Anschließend öffnet sich ein Fenster mit der Liste aller Dateien. Diese beinhaltet auch eine Auflistung aller Warnungen und Fehler, falls Dateien nicht erfolgreich importiert werden konnten (siehe Abbildung 6).

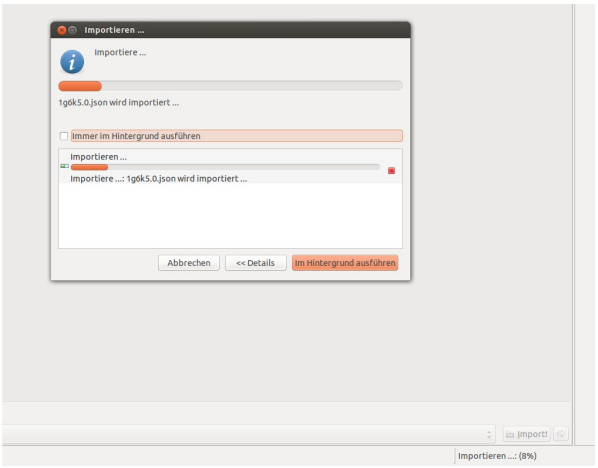

Abbildung 5: Kontext-Menü des Importvorgangs. Dieses erscheint nach Betätigung des Import-Buttons

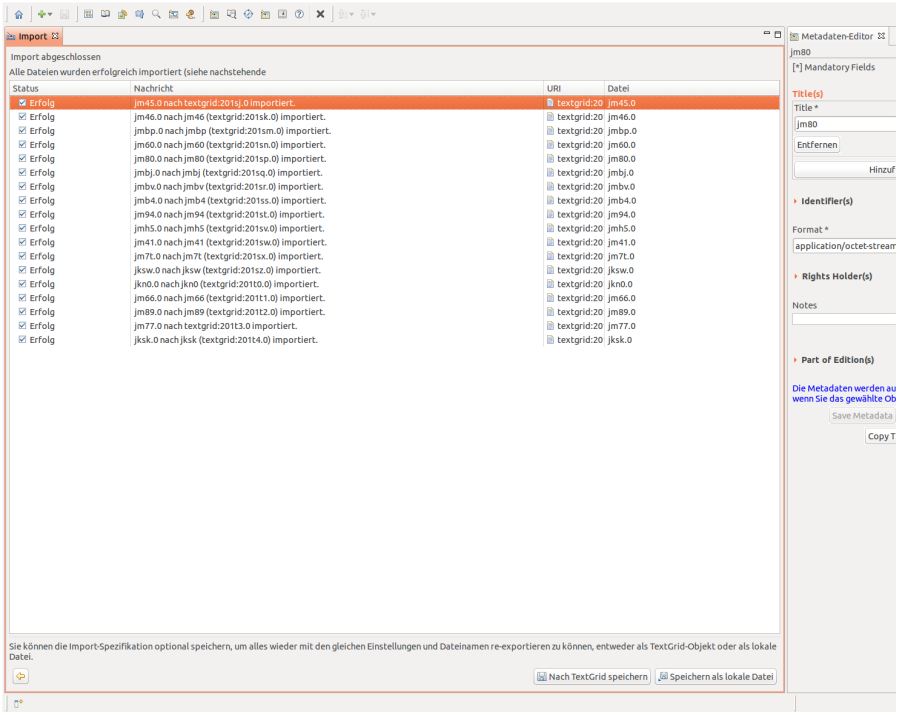

Abbildung 6: Auflistung über den Importvorgang aller Dateien mit entsprechendem Vermerk über den Erfolg des Vorgangs und der vergebenen Textgrid-URI

6. Die spezifischen Einstellungen für einen bestimmten Importvorgang können gespeichert werden, um einen späteren Re-Import der selben Objekte zu vereinfachen bzw. zu beschleunigen. Die Einstellungen beinhalten eine Liste der originalen Dateinamen zusammen mit den zugehörigen TextGrid URIs. Die Importspezifikationen können als TextGrid-Datei oder als lokale Datei gespeichert werden. Dafür können die Buttons auf der unteren Fenstersektion genutzt werden.

6.2 Re-Import von TextGrid-Objekten

Für im Vorhinein exportierte Dateien aus dem TextGridLab besteht die Möglichkeit, einen Re-Import dieser Objekte durchzuführen. Durch den Exportvorgang wurden die Datei selbst und eine zusätzliche Datei, welche die korrespondierenden Metadaten enthält, auf dem externen oder lokalen Speichermedium gespeichert.

Das spezifische Objekt kann durch die Auswahl in der Projekt-Dropdownbox im Importfenster entweder als neues Objekt mit einer neuen URI oder als neue Revision des vorherigen Objektes re-importiert werden. Beide Möglichkeiten werden im Folgenden beschrieben.

6.2.1 Re-Import einer Datei als neues Objekt

Bevor dieser Vorgang gestartet werden kann, muss das Objekt zuerst aus dem TextGrid-Lab exportiert werden. Die exportierten Objekte werden daraufhin auf dem individuellen lokalen Dateisystem gespeichert.

1. Im ersten Schritt muss das Import Tool über *Datei > Importieren von lokalen Dateien...* geöffnet werden.

2. Die Auswahl des Zielprojektes, in das die vorher exportierten Objekte re-importiert werden sollen, muss erfolgen. Im Anschluss können über einen Klick auf den *Hinzufügen* Button die entsprechenden Dateien auf dem lokalen Betriebssystem ausgewählt werden. Dabei können entweder die Dateien selbst oder auch die Dateien zusammen mit den korrespondierenden Metadaten ausgewählt werden. Durch einen Klick auf *Öffnen* werden die ausgewählten Dateien im Importfenster angezeigt. Eine weitere Möglichkeit des Hinzufügens besteht durch Drag & Drop.

   Zu jeder Datei, die zum Importfenster hinzugefügt wurde, werden Details (lokale Dateiname, TextGrid-URI und Dateiformat) angezeigt. Diese sind zu diesem Zeitpunkt noch nicht re-importiert und neue TextGrid-URIs sind ebenfalls noch nicht generiert. Durch Markieren der Datei im Importfenster und einen Links-Klick öffnen sich die Metadaten der ausgewählten Datei. Zu diesem Zeitpunkt können noch weitere Metadaten zu dem Objekt hinzugefügt werden oder der Titel geändert werden.

3. Durch einen Klick auf *Importieren* wird der Re-Importvorgang abgeschlossen.

6.2.2 Re-Import einer Datei als neue Revision des vorherigen Objektes

Bevor dieser Vorgang gestartet werden kann, muss das Objekt ebenfalls zuerst aus dem TextGridLab exportiert werden. Die exportierten Objekte werden daraufhin auf dem individuellen lokalen Dateisystem gespeichert.

1. Im ersten Schritt muss das Import Tool über *Datei > Importieren von lokalen Dateien...* geöffnet werden.

2. Die vorher exportierten Dateien müssen zum Importfenster hinzugefügt werden. Diese kann durch einen Klick auf den *Hinzufügen*-Button oder durch Drag & Drop erledigt werden.

3. In der Auswahlbox kann der originale Projektordner, aus dem die Dateien exportiert wurden, ausgewählt werden. Dabei ist es wichtig, dass sichergestellt ist, dass der Ordnername das Format *Projektname – Neue Revisionen* hat. Sobald der Ordner ausgewählt wurde, wird die vorherige TextGrid URI erkannt.

4. Nach dem Klick auf *Importieren* werden die Dateien als neue Revisionen im originalen Projekt gespeichert.

6.3 Import Tool Extern (koLibRI)

Das externe Import-Tool kann genutzt werden, um Objekte sowohl in das TextGridLab als auch in das TextGridRep zu importieren. Für diesen Importvorgang können verschiedene Strategien – sogenannte Policies – angewendet werden.

- Eine Möglichkeit besteht darin, die Dateien und Ordner in ein Hotfolder zu kopieren. Bei dieser Policy muss nichts TextGrid-spezifisches mit den Daten getan werden. Die notwendigen TexGrid-Metadatendateien werden automatisch aus dem Dateinamen erstellt. Die Ordnerstruktur wird durch die Nutzung von TextGrid-Aggregationen übernommen.

- Des Weiteren kann eine DFG-Viewer METS-Datei genutzt werden. Alle verlinkten Dateien werden dabei heruntergeladen und die Struktur der beiden StructMaps (logisch und physisch) werden als TextGrid-Aggregationen importiert.

- Sind bereits TextGrid-Metadatendateien vorhanden, ist es möglich, dass die Policy *complete_import* genutzt werden kann. Die erforderliche vorherige Erstellung der Metadatendateien mitsamt evtl. Referenzen setzt jedoch Kenntnisse des TextGrid Metadatenschemas und des TextGrid Objektkonzepts voraus und muss von den importierenden Nutzerinnen und Nutzern übernommen werden.

Diese drei Policies können genutzt werden, um Dateien in das TextGridLab und das TextGridRep zu importieren. Dafür müssen lediglich die entsprechenden Service-Endpoints konfiguriert werden. Der Unterschied zwischen einem Import in das TextGridLab und in das TextGridRep liegt in der Sichtbarkeit für die Nutzerinnen und Nutzer. Werden Dateien in das TextGridLab importiert, so sind diese nur für die Erstellerinnen und Ersteller und die für das Projekt freigegebenen Personen sichtbar. Innerhalb des TextGridLab kann dieser Nutzerkreis die Dateien lesen und ggf. bearbeiteten. Die Rolle des Projekt-Managers hat dann die Möglichkeit, die Daten mittels der TG-publish GUI zu veröffentlichen. Bei einem Import in das TextGridRep sollen die Dateien generell öffentlich zugänglich gemacht werden. Die Dateien werden jedoch zunächst nur in der TextGrid-Sandbox veröffentlicht, um diese dort noch einmal zu überprüfen und im Anschluss dann endgültig zu veröffentlichen.

In jedem Fall ist eine TextGrid Projekt-ID und eine RBAC Sitzungs-ID notwendig. Beide werden durch das TextGridLab bereitgestellt. Der Log-In in das TextGridLab funktioniert durch die Nutzung eines TextGrid-Accounts (TextGrid LDAP oder Shibboleth). Dort können Projekte erstellt und die Projekt-ID überprüft werden. Über das Menü *Hilfe* > *Authentifizierung* kann dort auch die Sitzungs-ID abgerufen werden.

6.3.1 Nutzungsmöglichkeiten

Für die folgenden Nutzungsmöglichkeiten sind die beiden Anwendungen Apache Maven[14] und Apache SVN[15] notwendig.

14 http://maven.apache.org/
15 http://subversion.apache.org/

Import durch SVN / Maven mit Eclipse

Als erstes muss das koLibRI TextGrid Add-on Modul heruntergeladen werden. Dies ist mittels des folgenden Konsolenbefehls möglich:

```
svn checkout
https://develop.sub.uni-goettingen.de/repos/koLibRI/tags/2012-08-29_koLibRI-t
extgrid-import_TG2.0 koLibRI-addon-textgrid-import
```

Daraufhin ist das Erstellen eines Eclipse-Projektes notwendig. Dies geschieht über den Befehl:

```
mvn eclipse:eclipse
```

Nun muss koLibRI konfiguriert werden. Siehe dazu die separierte Konfigurationssektion weiter unten.

Innerhalb der Eclipse-Anwendung muss nun das heruntergeladene Eclipse-Projekt koLibRI-addon-textgrid importiert werden. Durch den Eclipse Menü-Eintrag unter:

```
File > Import > Existing Project Into Workspace
```

Im Anschluss kann koLibRI gestartet werden. Dafür ist es notwendig, eine Eclipse Runtime-Konfiguration zu erstellen:

- Erstellen einer neuen Java Applikation im Eclipse Runtime Menü
- Als Hauptklasse muss de.langzeitarchivierung.koLibRI.WorkflowTool gewählt werden
- Hinzufügen von -c [Pfad zur gewählten Konfigurationsdatei]
- Unter Umständen ist es notwendig, der Applikation mehr Arbeitsspeicher einzuräumen. Mittels des Befehls -Xmx1024m ist dies möglich.

Import durch SVN / Maven auf der Kommandozeile

Als erstes muss das koLibRI TextGrid Add-on Modul heruntergeladen werden. Die ist mittels des folgenden Konsolenbefehls möglich:

```
svn checkout
https://develop.sub.uni-goettingen.de/repos/koLibRI/tags/2012-08-29_koLibRI-t
extgrid-import_TG2.0 koLibRI-addon-textgrid-import
```

Daraufhin wird das Projekt kompiliert:

```
mvn package
```

Nun muss koLibRI konfiguriert werden. Siehe dazu die separierte Konfigurationssektion weiter unten.

Unter Angabe der gewünschten Konfigurationsdatei muss nun zum Starten von koLibRI folgender Befehl in die Konsole eingegeben werden:

```
mvn exec:java -Dexec.args="-c path/to/config_file.xml"
```

Import durch die koLibRI JAR Datei

Um die JAR-Datei nutzen zu können, muss zuerst eine Konfigurationsdatei heruntergeladen werden. Der folgende Link führt zu einer ZIP-Datei, die alle notwendigen Konfigurationsdateien und Templates enthält

```
https://develop.sub.uni-goettingen.de/repos/koLibRI/tags/2012-08-29_koLibRI-t
extgrid-import_TG2.0/src/main/resources/koLibRI-textgrid-import.zip
```

Zusätzlich ist das koLibRI Kommandozeilenmodul notwendig. Die folgende JAR-Datei muss im selben Ordner gespeichert werden wie die vorherige ZIP Datei.

```
https://repository.bibforge.org/archiva/repository/snapshots/de/langzeitarchi
vierung/koLibRI/koLibRI-cli/2.0-SNAPSHOT/koLibRI-cli-2.0-SNAPSHOT.jar
```

Nun sollte eine Ordnerstruktur ähnlich der Folgenden vorhanden sein:

```
koLibRI-textgrid-import
    | config
        | tglab_config.xml
        | tgrep_config.xml
        | policies.xml
        | ...some more koLibRI config files...
    | folders
        | dest
        | hotfolder
            | data
        | log
        | temp
        | work
    | koLibRI-cli-2.0-SNAPSHOT.jar
```

Nun muss koLibRI konfiguriert werden. Siehe dazu die separierte Konfigurationssektion weiter unten.

Ist die Anwendung korrekt konfiguriert und alle gewünschten Dateien kopiert, kann koLibRI gestartet werden. Dafür ist eine Java-Virtual-Machine notwendig, die Java 6 oder höher benutzt. Um die JAR-Datei auszuführen, muss in den Ordner navigiert werden, der die koLibRI JAR-Datei enthält. Zum Starten wird folgender Befehl verwendet:

```
java -jar koLibRI-cli-2.0-SNAPSHOT.jar -c config/tglab_config.xml
```

7.3.2 Konfiguration von koLibRI[16]

Auswahl der Konfigurationsdatei

Im *config/* Ordner existieren zwei Konfigurationsdateien. Die Datei

16 https://dev2.dariah.eu/wiki/pages/viewpage.action?pageId=6783190

tglab_config.xml

wird genutzt, um Daten in TextGrid Lab zu importieren, sodass der Nutzer im ausgewählten TextGrid Projekt gearbeitet werden kann. Die Daten sind nur für die Erstellerinnen und Ersteller sichtbar und für die Nutzerinnen und Nutzer, denen der Zugriff auf die Daten gewährt wurde. Alle Service-Endpunkte für einen TG-lab-Import sind in dieser Konfigurationsdatei bereits vorkonfiguriert. Die Datei

tgrep_config.xml

wird für einen direkten Import in das TextGridRep genutzt. Danach sind die importierten Daten unmittelbar für alle öffentlich sichtbar. (zuerst nur in der Sandbox und nach finaler Publikation für alle).

Editierung der Konfigurationsdatei

<field>defaultPolicyName</field>

Setzen der Importstrategie: Der Parameter **defaultPolicyName** kann die folgenden Policies adressieren:

aggregation_import

Diese Policy wird genutzt, um automatisch TextGrid-Metadaten für jede Datei zu erstellen (aus dem Dateinamen und dem Dateiformat). Für jeden Ordner wird eine TextGrid-Aggregation erstellt und importiert. Dadurch ist nachher die Ordnerstruktur in TextGrid dieselbe wie im Importordner.

complete_import

Mit dieser Policy werden alle Dateien Importiert. Dabei werden keine zusätzlichen Metadaten erstellt, sodass ein kompletter Satz von TextGrid-Objekten inklusive den entsprechenden TextGrid-Metadaten erstellt werden muss, die außerdem schon mit TextGrid-URIs versehen sind. Dateierweiterungen für existierende TextGrid Editionen, Kollektionen, Werke, Aggregationen, XML-Dateien und Metadatendateien können bei Bedarf konfiguriert werden.

continue_import

Bei einem angehaltenen oder abgebrochenen Import ist diese Policy anzuwenden. Dafür ist es lediglich notwendig, den Hotfolder als temporären Ordner zu konfigurieren, in dem die Dateien bearbeitet werden.

delete_import

Ein bereits importierter Satz an Objekten kann aus der Sandbox wieder gelöscht werden. Dieser Prozess kann durch eine Liste mit URIs oder durch die Angabe einer Root URI durchgeführt werden.

publish_import

Mittels dieser Policy kann ein bereits importierter Objektsatz in die Sandbox des das TextGridRep endgültig veröffentlicht werden.

dfgviewermets_import

Nimmt als Input eine (oder mehrere) DFG-Viewer METS Dateien nach DFG-Viewer-Spezifikation und erstellt eine Ordnerstruktur anhand der physikalischen und logischen StructMap, die dann in TextGrid importiert wird. MODS und/oder TEI Metadaten werden gegen die TextGrid Metadaten gemappt mittels existierender MODS/TEI XSL Tranformationsdateien. Dieser Prozess kann auch mit einer selbst erstellten XSL Transformationsdatei geschehen.

<field>tgcrudServerUrl</field>

Mittels dieses Feldes kann der Endpunkt des TG–-crud Services gewählt werden. Abhängig von der Wahl der Import Location (TG-Lab oder TG.-Rep) ist der TG-crud-Endpoint bereits konfiguriert.

<field>rbacSessionId</field>, <field>projectId</field>

Diese beiden Felder sind für die Einstellungen der Authentifikation und des Projektes wichtig. Dafür müssen die beiden Felder mit den Werten der entsprechenden Projekt ID und der Session ID (**rbacSessionId**) gefüllt werden.

<field>logParameter</field>

Zur Zeit wird diese Funktion in textGrid nicht genutzt, daher kann das Feld leer gelassen werden.

<field>getPids</field>

Ist das Feld getPids auf TRUE gesetzt, werden für alle zu importierenden TextGrid-Objekte PIDs generiert. Dafür wird der GWDG Handle Service genutzt. Dieser Vorgang ist nur in dem Fall sinnvoll, wenn Daten direkt in das TextGridRep importiert werden.

Aggregation Import Konfiguration

Wird der *aggregation_import* genutzt, müssen die Daten wie beschrieben konfiguriert und im Anschluss koLibRI gestartet werden.

<field>hotfolderDir</field>

Nutzung eines *Hotfolders:* Als Hotfolder ist *./folders/hotfolder/data/* vorkonfiguriert. Um die Daten publizieren zu können, müssen die Daten nur in diesen Ordner kopiert werden. Vor dem Start werden die Daten kopiert, sodass die Originaldaten bestehen

bleiben. Die erste Aggregation wird der Unterordner in *data/* (nur für den *aggregation_import*).

<field>useBaseUrisInAggregations</field>

Base URIs: Die letzte Möglichkeit besteht in der Nutzung von Base URIs (textgrid:1234) oder absoluten URIs (*textgrid:1234.0*). Dies ist nur möglich, wenn der *aggregation_import* genutzt wird und wenn die BaseUris in den generierten Aggregationen vorhanden sind.

Complete import Konfiguration

Wird der *aggregation_import* genutzt, müssen die Daten wie beschrieben konfiguriert und im Anschluss koLibRI gestartet werden.

<field>hotfolderDir</field>

Nutzung eines *Hotfolders:* Als Hotfolder ist *./folders/hotfolder/data/* vorkonfiguriert. Um die Daten publizieren zu können, müssen die Daten nur in diesen Ordner kopiert werden. Vor dem Start werden die Daten kopiert, sodass die Originaldaten bestehen bleiben. Alle Daten werden genau gemäß den getroffenen Vorbereitungen entsprechend importiert.

DFG-Viewer METS import Konfiguration

Wird der *dfgviewermets_import* genutzt, müssen die Daten wie beschrieben konfiguriert werden und im Anschluss koLibRI gestartet werden.

<field>hotfolderDir</field>

Nutzung eines *Hotfolders:* Als Hotfolder ist *./folders/hotfolder/data/* vorkonfiguriert. In diesen Ordner müssen METS-Daten platziert werden. Für jede METS-Datei wird eine Root-Aggregation/Edition/Kollektion erstellt. Es ist möglich mehr als eine METS-Datei in das Hotfolder zu platzieren. In diesem Fall wird der Import mit parallel laufenden konfigurierbaren Threads durchgeführt.

<field>rootAggregationMimetype</field>

DFG Viewer Aggregationen: Für den Import über den DFG-Viewer kann das Format der Root-Aggregation gewählt werden (Für jede METS Datei existiert eine Root-Aggregation). Es kann gewählt werden, ob die Daten als TextGrid-Aggregation (**text/tg.aggregation+xml**), Edition (**text/tg.edition+tg.aggregation+xml**) oder Kollektionen (**text/tg.collection+tg.aggregation+xml**) importiert werden sollen.

Publish configuration

Um die Objekte endgültig zu publizieren (nur nach der Veröffentlichung in der Sandbox), muss die Policy **publish_import** genutzt werden.

<field>hotfolderDir</field>

Der **hotfolderDir** Wert muss zum temp-Ordner des Imports geändert werden, um die importierten Daten wieder löschen zu können (z.B. *./folders/temp/1318521646580_data/*.). Die Verwendung von absoluten Pfaden funktioniert ebenfalls. Die entsprechenden URIs werden von der erstellen URI mapping Datei (**.IM.tsv)* entnommen.

Löschkonfiguration

Bereits veröffentlichte Daten können gelöscht werden, wenn sie in die TextGrid -Sandbox importiert wurden. Dafür muss die Policy zu **delete_import** geändert werden.

<field>deleteViaMappingFile</field>

Der Wert dieses Feld muss auf TRUE gesetzt werden, wenn eine Mapping-Datei für das Löschen genutzt werden soll. Der Wert des Felds *hotfolderDir* muss ebenfalls entsprechend gesetzt werden. Ist der Wert FALSE eingetragen, wird die URI des Root-Objektes genutzt. Dabei sollte der Vorgänger gelöscht werden. In diesem Fall muss die rootUri gesetzt werden.

<field>hotfolderDir</field>

Der **hotfolderDir** Wert muss zum temp-Ordner des Imports geändert werden, um die importierten Daten wieder löschen zu können (z.B. *./folders/temp/1318521646580_data/*.). Die Verwendung von absoluten Pfaden funktioniert ebenfalls. Dabei sei angemerkt, dass die Angabe des absoluten Pfades zur Mapping-Datei selbst nicht funktioniert. Die entsprechenden URIs werden von der erstellen URI mapping Datei (**.IM.tsv)* entnommen.

<field>rootUri</field>

Die Root-URI (z.B.: textgrid:1234.0) um alle Objekte zu löschen, die durch diese Root-Aggregation referenziert werden. Dieser Vorgang läuft rekursiv ab.

7. Dokumentation des Imports von Satellitenprojekten

Der Datenimport in TextGrid soll exemplarisch an den beiden Satellitenprojekten, der Edition von Theodor Fontanes Notizbüchern und Blumenbach-online, gezeigt werden. Dabei werden zum einen Probleme deutlich, die während des Importvorgangs aufgetreten sind, zum anderen wird sichtbar, welche grundlegenden Voraussetzungen dafür nötig waren.

7.1 Hybrid-Editon von Theodor Fontanes Notizbüchern

7.1.1 Allgemeines

Ziel des Projekts ist die Erarbeitung und Publikation einer genetisch-kritischen und kommentierten Hybrid-Edition, die erste vollständige und material- und medienbasierte Edition der insgesamt 67 überlieferten Notizbücher Fontanes. Die Edition entsteht unter der Projektleitung von Dr. Gabriele Radecke und der informationswissenschaftlichen Leitung von Dr. Heike Neuroth an der Theodor Fontane-Arbeitsstelle des Seminars für Deutsche Philologie (Universität Göttingen) in Zusammenarbeit mit der Niedersächsischen Staats- und Universitätsbibliothek Göttingen. Assoziierte Partnerin ist die Staatsbibliothek zu Berlin, Preußischer Kulturbesitz, die als Eigentümerin der Notizbücher das Urheberrecht an den Digitalisaten innehat. Das textkritische und philologische Editionsprojekt wird durch digitale Methoden und die Virtuelle Forschungsumgebung TextGrid maßgeblich unterstützt. Ein Teil des Workflows wird auf der Basis von TextGrid-Anwendungen ausgeführt.

7.1.2 Datenbestand

Es sind insgesamt 67 Notizbücher überliefert. Die Anzahl an Scans muss von der tatsächlichen Blatt- bzw. Seitenzahl unterschieden werden. Zusätzlich zu den beschrifteten und unbeschrifteten Blättern/Seiten wurden die Einbände und Buchrücken, die auf- und angeklebten Blätter, die Beilagen und pro Notizbuch einmal die auf dem Buchrücken aufliegende Farbskala aufgenommen. Hinzu kommen Beilagen und Scans auch von nur teilweise enthaltenen Seiten, die herausgerissen oder ausgeschnitten wurden. Beilagen und angeklebte Materialien wurden, sofern möglich in mehreren Positionen aufgenommen, so ist eine aufgeklebte Tasche zum Beispiel geöffnet und geschlossen abgebildet. Es ergeben sich zum Beispiel für das Notizbuch „D10“ 51 Scans und für Notizbuch „E06“ 186 Scans. Der Boxplot (Abbildung 5) verdeutlicht die Streuung der Scananzahl ohne Berücksichtigung evtl. Leerseiten.

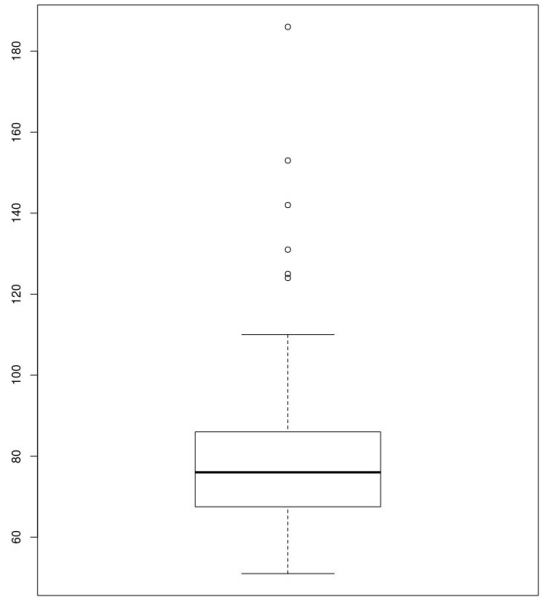

Abbildung 7: Boxplot Seitenanzahl in den Notizbüchern

Zu Forschungszwecken stehen dem Projekt TIFF-Images zur Verfügung. Im Rahmen der Edition werden komprimierte Versionen im JPEG-Format publiziert. Die Notizbücher wurden für das Editionsprojekt neu digitalisiert, da die Erstdigitalisierung lediglich aus archivalischen Gründen erfolgte, die in Bezug auf Qualität und Quantität nicht den editionswissenschaftlichen und philologischen Anforderungen einer material- und medienorientierten Notizbuch-Edition genügte. So waren teilweise nur retrodigitalisierte Aufnahmen (Quelle: Mikrofilm) verfügbar und viele Seiten (u. a. Blattfragmente und Vakat-Seiten) nicht aufgenommen. Da auch diese Seiten eine hohe Relevanz haben, etwa weil sie Rückschlüsse auf die Chronologie der Blattbeschriftung und die Arbeitsweise Fontanes erlauben, wurde die editionsspezifische Neudigitalisierung veranlasst und von der Projektleiterin begleitet.

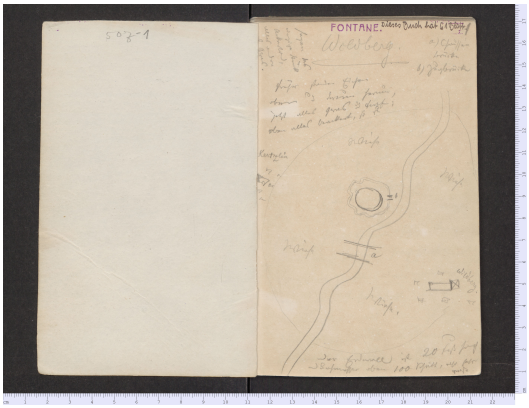

Abbildung 8: Fontane: Notizbuch A1 (Scan A01_002.jpg); © Staatsbibliothek zu Berlin, Preußischer Kulturbesitz

Insgesamt wurden 5427 Scans als TIFF-Formate erstellt und auf einem externen Datenträger abgespeichert, der dem Projekt zur Verfügung steht. Die Festplatte enthält pro Notizbuch ein Verzeichnis. Das JPEG-Format wurde von einem Projektmitarbeiter im Zuge des Ingest erzeugt.

7.1.3 Beispieldaten

TIFF-Format

Auflösung:

Width: 5563px
Height: 4327px

Dateigröße: 72,9 MB (72.882.739 bytes)

TextGrid-Metadaten-Objekt:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<object xmlns="http://textgrid.info/namespaces/metadata/core/2010"
xmlns:tg="http://textgrid.info/relation-ns#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <generic>
        <provided>
            <title>A01_002.tif</title>
            <format>image/tiff</format>
        </provided>
        <generated>
            <created>2012-10-09T14:36:07.036+02:00</created>
            <lastModified>2012-10-09T14:36:07.036+02:00</lastModified>
            <textgridUri extRef="">textgrid:164r7.0</textgridUri>
            <revision>0</revision>
            <extent>72882739</extent>
            <dataContributor>stefan.funk@textgrid.de</dataContributor>
```

```
            <project id="TGPR-9b18459d-3a6b-004f-b6aa-4fba2b9e1d3e">Fontane
Notizbücher</project>
            <permissions>delete read write</permissions>
        </generated>
    </generic>
    <item>
        <rightsHolder id="">Staatsbibliothek zu Berlin - Preußischer Kulturbesitz,
Handschriftenabteilung</rightsHolder>
    </item>
</object>
```

Weitere EXIF- oder Metadaten sind nicht elektronisch übermittelt.

JPEG-Format

Auflösung:

Width: 5563px
Height: 4327px

Dateigröße: 2,6 MB (2.595.499 bytes)

TextGrid-Metadaten-Objekt:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<object xmlns="http://textgrid.info/namespaces/metadata/core/2010"
xmlns:tg="http://textgrid.info/relation-ns#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <generic>
        <provided>
            <title>A01_002.jpg</title>
            <format>image/jpeg</format>
        </provided>
        <generated>
            <created>2012-10-09T14:35:02.428+02:00</created>
            <lastModified>2012-10-09T14:35:02.428+02:00</lastModified>
            <textgridUri extRef="">textgrid:164n4.0</textgridUri>
            <revision>0</revision>
            <extent>2595499</extent>
            <dataContributor>stefan.funk@textgrid.de</dataContributor>
            <project id="TGPR-9b18459d-3a6b-004f-b6aa-4fba2b9e1d3e">Fontane
Notizbücher</project>
            <permissions>delete read write</permissions>
        </generated>
    </generic>
    <item>
        <rightsHolder id="">Staatsbibliothek zu Berlin - Preußischer Kulturbesitz,
Handschriftenabteilung</rightsHolder>
    </item>
</object>
```

7.1.4 Importvorgang

Zum Import wurde koLibRI eingesetzt und dabei auf die hotfolder-Methode zurückgegriffen. Die Konfiguration wurde entsprechend den vorstehenden Möglichkeiten angepasst.

Zum Importvorgang genügt es ein Ausgangsverzeichnis anzugeben und die geforderten Konfigurationsdateien zu hinterlegen. Die darin befindlichen Daten werden in einen

temporären Ordner kopiert. In unserem Fall wurden keine Aggregationen erstellt. Anschließend werden entsprechend der Dateianzahl TextGrid-URIs erzeugt.

Auszug aus der Logdatei:

```
<RUNNING>  hotfolder  -->
<RUNNING>  hotfolder  -->  Files are beeing counted
<RUNNING>  hotfolder  -->  Found 0 folders and 134 files to process
<RUNNING>  hotfolder  -->  Requesting total of 134 URIs
<RUNNING>  hotfolder  -->  URIs generated: textgrid:164mb...textgrid:164rx
<DONE>     hotfolder  -->  134 TextGrid URIs created in 2 seconds 429 milliseconds
<RUNNING>  hotfolder  -->  Initialized ActionModule
de.langzeitarchivierung.koLibRI.actionmodule.MetadataGenerator
<RUNNING>  hotfolder  -->  Initializing JHOVE
<RUNNING>  hotfolder  -->  Creating technical metadata
```

koLibRI greift zur Erstellung der technischen Metadaten auf JHOVE zurück. Damit werden die Content-Types erstellt, die als Metadatum in die technischen Metadatenobjekte einfließen. In diesem Schritt wird für jede Datei eine eigene JHOVE-Datei erstellt, die den entsprechenden Output beinhaltet. Damit verdoppelt sich die Dateianzahl in der hotfolder.

Auszug aus der Logdatei:

```
<RUNNING>  hotfolder  -->  Stored JHOVE technical metadata file:
/koLibRI-addon-textgrid-import/./folders/temp/1337871454359_hotfolder/A01_002.jpg.jh
ove
<DONE>     hotfolder  -->  Metadata successfully created in 52 seconds
<RUNNING>  hotfolder  -->  Initialized ActionModule
de.langzeitarchivierung.koLibRI.actionmodule.textgrid.TextgridMetadataProcessor
<RUNNING>  hotfolder  -->  URI list contains 134 URIs
<RUNNING>  hotfolder  -->  Metadata is beeing created for all files in:
/koLibRI-addon-textgrid-import/./folders/temp/1337871454359_hotfolder
<RUNNING>  hotfolder  -->  Metadata template file loaded:
./config/fontane_tiff_metadata_template.xml
<RUNNING>  hotfolder  -->  Found 0 folders and 268 files to process
<RUNNING>  hotfolder  -->  [A01_002.tif] Title set from filename
<RUNNING>  hotfolder  -->  [A01_002.tif] Format set to: image/tiff
<DONE>     hotfolder  -->  TextGrid metadata created in 1 second 187 milliseconds
```

Anschließend wird die Übertragung der Daten gestartet und pro Datei eine URI zugewiesen.

Auszug aus der Logdatei:

```
<RUNNING>  hotfolder  -->  Initialized ActionModule
<RUNNING>  hotfolder  -->  Starting to create aggregation files
<DONE>     hotfolder  -->  All aggregations have been sucessfully created in 6
milliseconds
<RUNNING>  hotfolder  -->  Initialized ActionModule
de.langzeitarchivierung.koLibRI.actionmodule.textgrid.RenameAndRewrite
<RUNNING>  hotfolder  -->  URI list contains 134 URIs
<RUNNING>  hotfolder  -->  Base path set to:
/koLibRI-addon-textgrid-import/folders/temp/1337871454359_hotfolder
<RUNNING>  hotfolder  -->  Creating import map
<RUNNING>  hotfolder  -->  [textgrid:164r7.0] Import entry added: textgrid:164r7.0 =
/koLibRI-addon-textgrid-import/folders/temp/1337871454359_hotfolder/A01_002.tif
<RUNNING>  hotfolder  -->  [textgrid:164n4.0] Import entry added: textgrid:164n4.0 =
/koLibRI-addon-textgrid-import/folders/temp/1337871454359_hotfolder/A01_002.jpg
```

Nun werden die Metadatenobjekte lokal erzeugt. Dazu wird das in einem vorigen Schritt erstellte Import-Mapping ausgewertet.

Auszug aus der Logdatei:

```
<RUNNING>  hotfolder  -->  Import mapping serialized to file:
/koLibRI-addon-textgrid-import/./folders/temp/1337871454359_hotfolder.IM.tsv
<RUNNING>  hotfolder  -->  Renaming files
<RUNNING>  hotfolder  -->  [textgrid:164r7] Files successfully renamed to:
164r7_A01_002.tif, 164r7_A01_002.tif.meta
<RUNNING>  hotfolder  -->  [textgrid:164n4] Files successfully renamed to:
164n4_A01_002.jpg, 164n4_A01_002.jpg.meta
<RUNNING>  hotfolder  -->  Rewriting pathes to TextGrid URIs
<RUNNING>  hotfolder  -->  [textgrid:164r7] Rewriting links in metadata file:
164r7_A01_002.tif.meta
```

Abschließend wird die Übertragung der Objekte mittels TG-crud angestoßen.

Auszug aus der Logdatei:

```
<DONE>     hotfolder  -->  All files have been renamed and rewritten in 1 second 351
milliseconds
<RUNNING>  hotfolder  -->  Initialized ActionModule
de.langzeitarchivierung.koLibRI.actionmodule.textgrid.GetPidsAndRewrite
<DONE>     hotfolder  -->  No PIDs shall be generated
<RUNNING>  hotfolder  -->  Initialized ActionModule
de.langzeitarchivierung.koLibRI.actionmodule.textgrid.SubmitFiles
<RUNNING>  hotfolder  -->  TG-crud web service WSDL file:
http://textgridlab.org/1.0/tgcrud/TGCrudService?wsdl
<RUNNING>  hotfolder  -->  Submitting all files in:
/koLibRI-addon-textgrid-import/./folders/temp/1337871454359_hotfolder
<RUNNING>  hotfolder  -->  [textgrid:164r7] Title: A01_002.tif, format: image/tiff,
size: 72882739 bytes
<RUNNING>  hotfolder  -->  [textgrid:164r7] TG-crud#CREATE complete
<RUNNING>  hotfolder  -->  [textgrid:164r7] Successfully deleted files from temp
folder
<RUNNING>  hotfolder  -->  [textgrid:164r7] -- 1/134 -- submission completed in 14
seconds 32 milliseconds
<RUNNING>  hotfolder  -->  [textgrid:164n4] Title: A01_002.jpg, format: image/jpeg,
size: 2595499 bytes
<RUNNING>  hotfolder  -->  [textgrid:164n4] TG-crud#CREATE complete
<RUNNING>  hotfolder  -->  [textgrid:164n4] Successfully deleted files from temp
folder
<RUNNING>  hotfolder  -->  [textgrid:164n4] -- 2/134 -- submission completed in 997
milliseconds
```

Auszug aus der Erfolgsmeldung:

```
<DONE>     hotfolder  -->  All files have been sucessfully submitted TO THE GRID in
38 minutes 59 seconds 52 milliseconds
<RUNNING>  hotfolder  -->  Initialized ActionModule
de.langzeitarchivierung.koLibRI.actionmodule.textgrid.Timelog
<RUNNING>  hotfolder  -->  Writing timelog file:
/koLibRI-addon-textgrid-import/./folders/temp/1337871454359_hotfolder/timelog.csv
<RUNNING>  hotfolder  -->  Timelog file:
de.langzeitarchivierung.koLibRI.actionmodule.textgrid.GetUris,2 seconds 429
milliseconds
de.langzeitarchivierung.koLibRI.actionmodule.textgrid.TextgridMetadataProcessor,1
second 187 milliseconds
de.langzeitarchivierung.koLibRI.actionmodule.textgrid.CreateAggregations,6
```

```
milliseconds
de.langzeitarchivierung.koLibRI.actionmodule.textgrid.RenameAndRewrite,1 second 351
milliseconds de.langzeitarchivierung.koLibRI.actionmodule.textgrid.SubmitFiles,38
minutes 59 seconds 52 milliseconds
<DONE>     hotfolder  -->  Timelog written
```

Damit ist der Import abgeschlossen.

7.1.5 Importberichte (Logfiles)

Die Importberichte haben für das Projekt eine große Bedeutung. Die TextGrid-URIs der Faksimiles werden im Attribut `facs` zum TEI-Element `surface` nicht genannt. Stattdessen ist der Attributinhalt der ehemalige Dateiname, der sich nun als Metadatum „Titel“ wiederfindet. Diese Anwendung dient der besseren (Menschen-) Lesbarkeit der XML-Daten, da die alphanummerische Sortierung der URIs nicht gleich der Sortierung der Titel ist und dadurch keinen Hinweise auf die Nummerierung der Digitalisate bietet.

Daher ist für eine spätere Weiterverarbeitung in Form einer Prozessierung – für eine evtl. synoptische Ansicht innerhalb des öffentlichen TextGrid-Repositorys unter Anwendung eines eigenen XSLT-Stylesheets – oder automatisierten Umbenennung der in ein eigenes Webportal exportierten Daten zwingend notwendig. Letzteres wurde nach vollständiger Übertragung auf eine projekteigene Virtuelle Maschine durchgeführt. Zudem ist die Weiterverarbeitung zum Beispiel in der Anwendung „digilib“ durch die Sortierbarkeit der Daten vereinfacht, da der automatisch generierte Bildkatalog nur auf Basis der Dateinamen sortiert.

7.2 Johann Friedrich Blumenbach – online

7.2.1 Allgemeines

Das an der Göttinger Akademie der Wissenschaften angesiedelte Projekt „Johann Friedrich Blumenbach – online“[17] hat sich zum Ziel gesetzt, die Werke Blumenbachs zu erschließen und die Projektergebnisse online zur Verfügung zu stellen. Dabei sollen nicht nur die Volltexte, sondern auch die in Blumenbachs Werken genannten Sammlungsobjekte digital erfasst werden, so dass interaktive Verweise zwischen digitalisierten Büchern und referenzierten Objekten möglich werden. Zur Unterstützung des Vorhabens wird die Virtuelle Forschungsumgebung TextGrid[18] eingesetzt. Im TextGridLab sollen z.B. die TEI-Volltexte und Digitalisate bearbeitet und mit Sammlungsobjekten verknüpft werden.

7.2.2 Datenbestand

Im Blumenbach Projekt werden TEI-Transkriptionen der Digitalisate mit Verweisen auf die Originalscans erstellt. Diese sollen erstmal in ein privates Projekt im TextGrid Repository eingespielt werden, um im TextGridLab weiter bearbeitet werden zu können. Aufgrund der Menge der Daten und der besseren Anpassbarkeit bei der Metadatenerstellung, wurde ein Ingest mit koLibRI dem über das TextGridLab vorgezogen. So besteht die Modelledition derzeit aus ca. 70 TEI-Dateien mit annähernd 23.000 Seitenscans als Bilddateien. Diese

17 http://www.blumenbach-online.de/
18 http://textgrid.de/

können besser von einem BatchJob auf dem Blumenbach-Server als über das TextGridLab eines Anwenders eingespielt werden. Desweiteren sollen die im TEI-Header hinterlegten Informationen genutzt werden, um beim Import direkt passende Editionen und Werke anzulegen und die Einzelobjekte diesen zuzuordnen.

Um das zu realisieren, wurde für koLibRI ein TEI Import-Modul geschrieben, welches mit einer Kombination aus TEI- und TextGrid-spezifischen Java-Klassen und XSLT-Stylesheets[19] die Daten für den Import vorbereitet, dabei notwendige Metadaten anlegt und dann alles in das Repository einspielt.

Als erstes wird hierbei aus dem TEI-Header der Uniform-Title ausgelesen und danach über eine TG-search Anfrage festgestellt, ob im Zielprojekt schon ein Werk mit entsprechendem Titel vorhanden ist. Falls ja, wird die entsprechende TextGrid-URI in die Metadaten der Edition eingetragen, falls das entsprechende Werk noch nicht vorhanden ist, wird es erstellt. Die Metadaten für Werk, Edition und Item kommen somit komplett aus dem TEI Header.

7.2.3 Vorgehen

Die TEI-Texte werden mit zugehörigen Bildern nacheinander in einen eigenen Unterordner des Hotfolders gelegt. Paralleles Anlegen mehrerer Jobs wäre generell auch möglich, aber da nur ein Werk pro Projekt angelegt werden soll, könnte es Probleme geben, wenn sowohl Job1 wie Job2 zum selben Werk gehören, beide beim Start noch keines vorfinden und dann dieses jeweils neu anlegen.

Eingehende Ordnerstruktur in *folders/hotfolder/data/*:

```
job1/0000001.xml
job1/images/000000001
job1/images/000000002
…
```

Aus der XML-Datei werden die Metadaten aus dem TEI-Header ausgelesen und dann entsprechende Werk-, Edition- und Item-Metadaten erstellt. Bei den Bilddateien wird mithilfe des in koLibRI eingebauten JHOVE-Moduls der Mime-Type extrahiert und es werden Metadaten erstellt, die als Titel den Dateinamen und den mit JHOVE erkannten Mime-Type als Format setzen. Das hat leider die Einschränkung, dass nur von JHOVE unterstützte Formate[20] erkannt werden, bei einem Format *image/png* zum Beispiel wird ein Objekt mit Format *application/octet-stream* erstellt.

Für den Unterordner *images* wird eine entsprechende Aggregation mit dem Titel „images“ erstellt. Der Link-Rewriter, der im Laufe der Prozessierung durch koLibRI läuft, ersetzt die Verweise auf Bilder in den TEI-Dokumenten wie z.B.

```
<pb n="9" facs="images/00000017" xml:id="pb009_0001">
```

19 Die XSLT Dateien sind einsehbar unter https://develop.sub.uni-goettingen.de/repos/kolibri/trunk/kolibri-addon-textgrid-import/config/transformation/blumenbach_tei/

20 http://jhove.sourceforge.net/documentation.html

des *facs* Attributs durch die entsprechende neue TextGrid-URI. Dabei werden vom Link-Rewriter auch referenzierte, aber nicht im Import enthaltene Objekte erkannt.

8. TextGrid-Metadaten

Metadaten werden in TextGrid an diversen Stellen verwendet, beispielsweise zur Beschreibung von Objekten, zur Beschreibung von Beziehungen zwischen Objekten und Sammlungen, sowie zur Verwaltung von Projekten und Benutzern. Im Folgenden ist eine Übersicht über die TextGrid-Metadaten gegeben; Ergänzungen und Verfeinerungen wurden parallel zur Entwicklung der Prototypen und des Testens vorgenommen.

8.1 Metadatenkategorien

In TextGrid werden zahlreiche Metadaten zur Beschreibung eines Objektes verwendet. Die einzelnen Metadaten lassen sich in Kategorien einteilen. Die im Anhang befindliche Abbildung 9 zeigt die Metadaten der einzelnen Kategorien.

Kategorie	**Beschreibung**
generated	Die Metadaten der Kategorie *generated werden* von der Middleware (TG-crud) erzeugt und mit jedem Aufruf von TG-crud ausgeliefert, welcher ein Metadaten Objekt sendet.
BibliographicCitationType	Metadaten, die von der Client-Anwendung erzeugt werden.
ObjectCitationType	Metadaten, die spezifisch für eine bestimmte Edition eines Objektes sind.
WorkType	Metadaten, die das Werk beschreiben.
ItemType	Die Metadatenkategorie *item* beschreibt ein erstelltes Objekt.
EditionType	Sammlung-Konzeptions abhängige Metadaten und dessen Beschreibung im public Bereich.
CollectionType	Eine Kollektion ist eine besondere Form der Sammlung|Aggregation, die einen besonderen Metadatensatz hat.
SourceType	Beziehung zwischen dem TextGrid Objekt und der Beschreibung der Quelle des Objektes.
CustomType	Ein Metadatentyp, um die bestehenden Metadaten mit nutzerspezifischen zu erweitern.

Tabelle 1: Metadatenkategorien in Textgrid

8.2 Metadatenstruktur

Die Struktur der Metadaten ergibt sich aus dem Aufbau der Metadaten im XML Format. Das oberste Element (Root-Element) ist das Objekt-Element und umschließt das gesamte Objekt und damit alle darauf folgenden Metadaten zur Beschreibung dieses Objektes.

Auf das Objekt-Element folgt das Element *generic*, dieses umfasst die Metadaten zu den von der Middleware TG-crud bereitgestellten Metadaten und die durch die Client-Anwendung erstellten Metadaten, also die beiden Elemente *provided* und *generated.*

Eine Beispiel XML-Datei zu dem Text „Etablissement Ronacher“ ist im Folgenden angegeben.

```
<?xml version="1.0" encoding="UTF-8"?>
<metadataContainerType xmlns:ns2="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:ns3="http://textgrid.info/namespaces/metadata/core/2010">
    <ns3:object>
        <ns3:generic>
            <ns3:provided>
                <ns3:title>Etablissement Ronacher</ns3:title>
                <ns3:format>text/tg.work+xml</ns3:format>
            </ns3:provided>
            <ns3:generated>
                <ns3:created>2011-12-16T20:31:23.264+01:00</ns3:created>
                <ns3:lastModified>2012-05-10T07:29:05.451+02:00</ns3:lastModified>
                <ns3:issued>2011-12-16T20:31:23.264+01:00</ns3:issued>
                <ns3:textgridUri extRef="">textgrid:jkk2.0</ns3:textgridUri>
     <ns3:revision>0</ns3:revision>
                <ns3:pid
pidType="handle">hdl:11858/00-1734-0000-0001-D833-9</ns3:pid>
                <ns3:extent>38</ns3:extent>
                <ns3:dataContributor>tvitt@textgrid.de</ns3:dataContributor>
                <ns3:project id="TGPR-372fe6dc-57f2-6cd4-01b5-2c4bbefcfd3c">Digitale
Bibliothek</ns3:project>
                <ns3:permissions>read</ns3:permissions>
                <ns3:availability>public</ns3:availability>
            </ns3:generated>
        </ns3:generic>
        <ns3:work>
            <ns3:agent role="author" id="pnd:118502255">Altenberg, Peter</ns3:agent>
            <ns3:dateOfCreation notBefore="1859" notAfter="1919"/>
            <ns3:genre>prose</ns3:genre>
        </ns3:work>
    </ns3:object>
</metadataContainerType>
```

8.3 TEI-XML Daten

Für die Speicherung der Volltextdaten verwendet TextGrid das TEI[21]-Format. Mit diesem ist ein Quasi-Standard in den Geisteswissenschaften und somit ein Synergie-Effekt zwischen den Anwendern und dem technischen Personal gegeben.

Um die Zugehörigkeit zwischen den Metadaten und dem Volltext darzustellen, wird für beide Dateien dieselbe TextGrid-URI verwendet.

21 http://www.tei-c.org/index.xml

9. Import großer Datenmengen

Für den Import großer Datenmengen sind vorherige Absprachen mit dem TextGrid-Konsortium notwendig. Diese Absprachen betreffen unter anderem den benötigten Speicherplatz für die zu importierenden Daten, da dieser im Abgleich mit dem verfügbaren Speicherplatz stehen muss. Aber auch die zu importierenden Dateitypen sind dabei von Relevanz, sowie die benötigten Metadaten. Des Weiteren ist es wichtig zu klären, ob die Daten zunächst in die Sandbox importiert werden sollen oder nicht.

Ein Beispiel für einen bereits getätigten Import einer großen Datenmenge ist das Archiv des Projektes *Digitale Bibliothek*[22]. Dieser Datenbestand von über 700.000 Dateien wurde bereits erfolgreich in das TextGrid-Repository importiert.[23]

22 http://www.deutsche-digitale-bibliothek.de/
23 http://textgridrep.de/repository.html

10. Tabellen und Abbildungen

11. Anhang

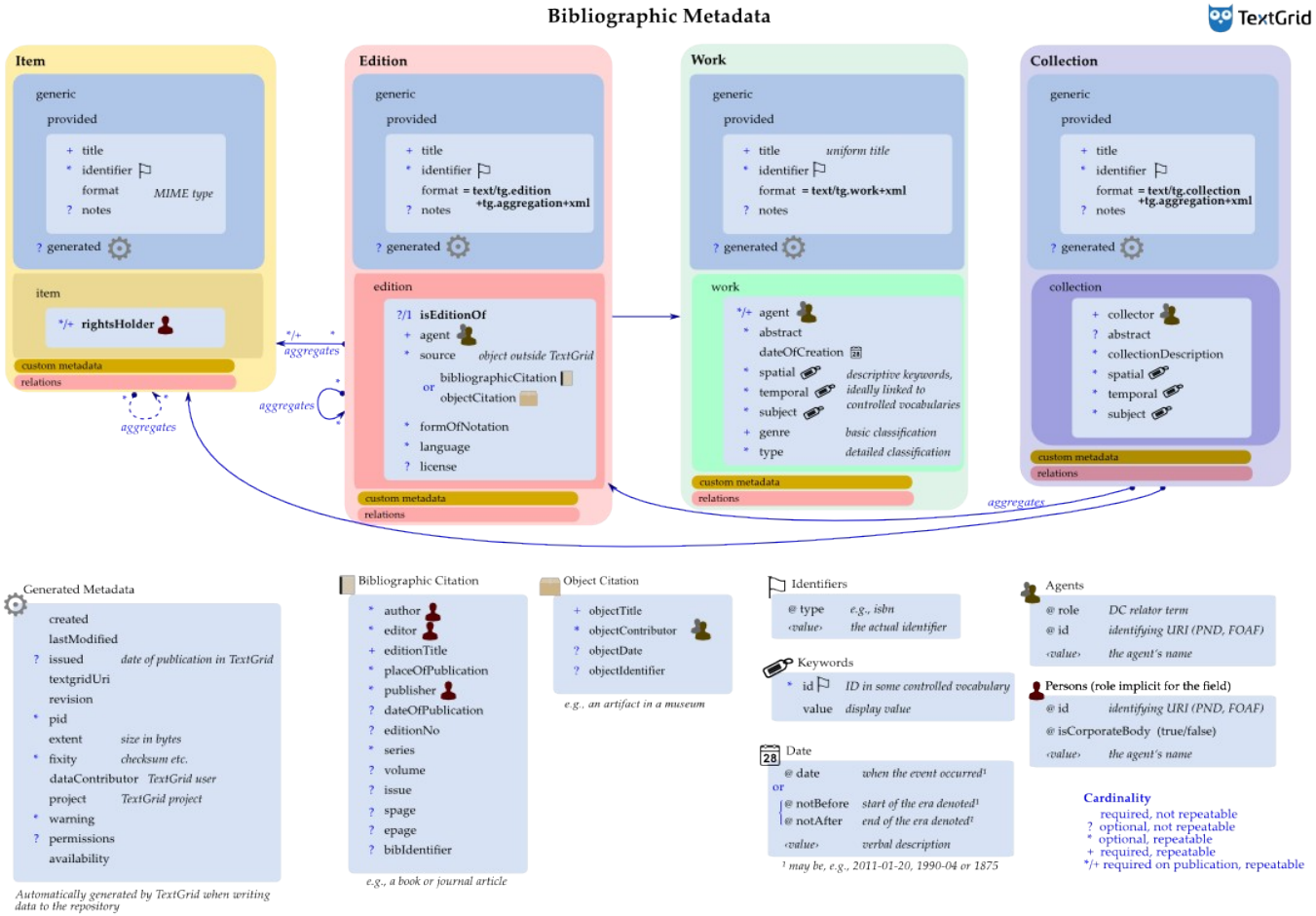

Abbildung 9: Übersicht über das TextGrid Metadatenschema