OLA-HD Konzept

Ein OCR-D Langzeitarchiv für historische Drucke

Langzeitarchivierung und persistente Identifizierung von OCR-Objekten

vorgelegt durch die
Niedersächsische Staats- und Universitätsbibliothek Göttingen

und die

Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen

This work is licensed under a Creative Commons Attribution 4.0 International License

Autoren:

SUB: Jörg-Holger Panzer, Zeki Mustafa Doğan, Kristine Schima-Voigt, Stefan Strathmann

GWDG: Prof. Dr. Philipp Wieder, Triet Ho Anh Doan

1 Einleitung

Das Koordinierungsprojekt OCR-D[1] befasst sich mit OCR[2] Workflows und der Standardisierung von Prozessen und Metadaten im Bereich Texterkennung oder optische Zeichenerkennung. Nachdem in der ersten Projektphase von OCR-D Entwicklungsbedarfe für Verfahren der automatischen Texterkennung ermittelt wurden, geht es in den sechs Modulprojekten der aktuellen Projektphase (2018-2020) um die Ermittlung von Anforderungen, die Spezifikation von Eigenschaften, Schnittstellen und Formaten, die Evaluation bestehender, sowie prototypischen Entwicklung neuer (Teil-)Lösungen.

Das Modul 5 hat sich mit Fragen der Langzeitarchivierung und Persistenz beschäftigt. Für die Verbesserung von OCR-Prozessen und -Workflows im Kontext von OCR-D, oder allgemeiner betrachtet, für die weitere wissenschaftliche Nutzung der OCR-Ergebnisse, ist eine nachhaltige Archivierung und Identifizierung der Digitalisate, der bibliographischen Metadaten sowie der erschlossenen Volltexte und deren Versionen notwendig. Die Verfügbarkeit und die Zitierfähigkeit der OCR-Texte ist eine wichtige Voraussetzung für die Arbeit mit den Daten und die Überprüfbarkeit wissenschaftlicher Ergebnisse. Es werden Lösungen für die dauerhafte Identifizierung des gesamten Werkes oder seiner logischen Untereinheiten, wie z. B. Kapitel, beziehungsweise physischer Untereinheiten, wie z. B. Seiten, benötigt.

Im Projekt OCR-D steht die Verbesserung von OCR-Prozessen und -Workflows im Vordergrund. Für die Qualitätsverbesserung ist die wiederholte Prozessierung derselben Vorlage mit geänderter OCR-Konfigurationen (OCR Engine, Algorithmus, Trainingsprozesse- und status, technische Umgebung, etc.) erforderlich. Es entstehen dabei Versionen von Prozessierungergebnissen, die zu speichern sind und zur Beurteilung von Konfigurationen herangezogen werden, um Prozesse zu verbessern.

Das vorliegende schriftliche Konzept als Projektergebnis enthält eine Diskussion der Anforderungen, die Spezifikationen der Daten, eine Evaluierung bestehender Lösungen, die Dokumentationen zu den technischen, organisatorischen und wirtschaftlichen Aspekten der Umsetzung, sowohl bezüglich der Langzeitarchivierung (LZA) als auch der persistenten Identifizierung. Die Archivobjekte sind im Rahmen des Projektes mit Dokumenten aus der OCR Prozessierung im Kontext von OCR-D klar abgegrenzt und der Rahmen wird durch die Projektausschreibung festgelegt.

Das Konzept wurde prototypisch implementiert (das OLA-HD) und in der Staats- und Universitätsbibliothek Göttingen (SUB) für die Archivierung der im hauseigenen Digitalisierungszentrum - Göttinger Digitalisierungszentrum (GDZ) - erzeugten OCR Volltexte als Demonstrator eingesetzt. Die Implementierung des OLA-HD und der Betrieb erfolgt durch das Rechen- und IT-Kompetenzzentrum Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen (GWDG).

---

[1] DFG-gefördertes Projekt OCR-D (https://ocr-d.de). Koordinierte Förderinitiative zur Weiterentwicklung von Verfahren der Optical Character Recognition (OCR). Hauptziel des Projekts ist “die konzeptionelle und technische Vorbereitung der Volltexttransformation der VD”. VD: Verzeichnissen der im deutschen Sprachraum erschienenen Drucke des 16.–18. Jahrhunderts (VD16, VD17, VD18).

[2] Eng. Optical Character Recognition (OCR)

2 Begriffsabgrenzung und Einordnung

2.1 Datenverwaltungssysteme

Es ist davon auszugehen, dass die Daten der OCR-Prozessierung (sowohl Eingabe-, als auch Ergebnisdaten) bereits in einem Datenverwaltungssystem beim Datenproduzenten gespeichert werden. Im einfachsten Fall kann die Speicherung in einem Dateisystem oder einem Objektspeicher (z.B. S3) erfolgen, oder etwas strukturierter über ein institutionelles Repositorium oder Dokumentenmanagementsystem (DMS). Diese Speichersysteme unterscheiden sich in den Strukturierungsmöglichkeiten der Daten, den Möglichkeiten Metadaten abzubilden und für die Datenorganisation einzusetzen, oder ganz allgemein in ihrem Funktionsumfang. Die Bandbreite der Funktionen reicht von CRUD[3] Funktionalitäten bis zur Erstellung von abgeleiteten Objekten (z.B. PDF-Erzeugung) oder dem Datenexport nach bestimmten Kriterien (z.B. alle Werke die nach einem bestimmten Datum importiert wurden). Die Systeme erfüllen zumeist nur einen Teil der Anforderungen an ein digitales Langzeitarchiv und es wird i.d.R. aktiv auf diesen Daten gearbeitet, d.h. Daten ändern sich und es wird ggf. nur der letzte Stand der Daten gespeichert.

2.2 Anwendungssysteme

Die auf die Datenverwaltungssysteme aufbauenden Anwendungssysteme (z.B. ein institutionelles Portal) machen Daten sichtbar und verfügbar. Sie bilden möglicherweise die einzige Schnittstelle zu den Daten. Diese Systeme ändern sich häufig, die Stabilität der Zugriffspfade wird nicht garantiert. Wenn es die rechtlichen Rahmenbedingungen erlauben, werden diese Systeme u.U. abgeschaltet. Die Daten müssen dann als verloren angesehen werden.

2.3 Nachweissysteme und Katalogsysteme

Nachweissysteme und Katalogsysteme speichern nur Metadaten und können für die Recherche eingesetzt werden. Für den Zugriff auf die Daten speichern sie Referenzen auf andere Systeme (auf Präsentationssysteme oder ein Archiv). Wegen der Kenntnis der fachspezifischen Daten und Metadaten, und dem Fokus auf die Bereitstellung von und Suche über Metadaten, unterstützen sie die Suche i.d.R. sehr gut.

Um den Entwicklungsaufwand nicht zu wiederholen, sollten die Funktion der Nachweis- und Katalogsysteme, soweit möglich, in das Archiv integriert werden, oder der Nutzer führt die Suche über die genannten Systeme durch und nutzt gefundene IDs für den Archivzugriff. Das Archiv wird damit von der komplexen Aufgabe einer generischen, für alle Kontexte gültige Metadatenextraktion befreit und es kann sich auf seine Kernaufgaben konzentrieren. Eine Volltextsuche wird von Nachweis- und Katalogsysteme i.d.R. nicht unterstützt.

[3] Abk. CRUD steht für **C**reate, **R**ead, **U**pdate and **D**elete

2.4 Digitale Langzeitarchivierung

Bei der Archivierung geht es sehr allgemein zusammengefasst, um die unbefristete Aufbewahrung von Objekten - nicht zwangsläufig im digitalen Umfeld. Die Annahme bei einem digitalen Archiv ist, dass große Datenmengen in Anzahl und Umfang dauerhaft stabil, konsistent und unverändert gespeichert werden müssen, das gespeicherte Ressourcen eindeutig identifiziert, d.h. durch eine eindeutige ID dauerhaft voneinander unterschieden und über die ID auch wiedergefunden werden, und das ein berechtigter Zugriff auf die Daten sichergestellt ist. Ein Archiv impliziert die Langzeitarchivierung der Archivobjekte. Die digitale Archivierung fordert die Sicherstellung der

- Authentizität (Unverfälschtheit, Vollständigkeit),
- Integrität (technische Korrektheit, Eindeutigkeit und Widerspruchsfreiheit),
- Verfügbarkeit (Zugreifbarkeit und Schnittstellen),
- Vertraulichkeit (geschützter Zugriff auf Archivobjekte) und
- Sicherheit (Schutz des Archivs selbst).

Die Punkte werden im Anforderungsabschnitt konkretisiert (s. 3 Anforderungen). In OCR-D kommt hinzu, dass die Anforderungen auch für Versionen von Archivobjekten sicherzustellen sind.

Ein digitales Archiv dient - wie bereits erwähnt - der zeitlich unbefristeten Aufbewahrung von digitalen Ressourcen, d.h. auf unterster Ebene von Dateien. Diese Dateien werden jedoch immer zu größeren Einheiten zusammengefasst und es werden digitale Objekte (Archivobjekte), mit eigenen Metadaten und Strukturen gebildet und gespeichert.

Neben den genannten funktionalen Anforderungen, muss sich ein Archiv ergänzend mit nicht-funktionalen Aspekten und konzeptionellen Fragen befassen, wie z.B. mit

- dem rechtlichen Rahmen und formalen Verpflichtungen, z.B. Datenschutz, Urheber- und Nutzungsrechte, Lizenzen, Vorgaben von Förderern, interne Vorgaben zum Bestandserhalt, etc.,
- dem wirtschaftlichen Rahmen, um die dauerhafte Finanzierung von Betrieb und Weiterentwicklung sicherstellen zu können,
- der Festlegung von Auswahl- und Aussonderungskriterien, Prioritäten, sowie von dafür erforderlichen Bewertungskriterien (Preservation Planning),
- der Sicherung der Interpretierbarkeit (Zukunftsfähigkeit der Daten),
- der Verfolgung der technischen Weiterentwicklung, oder mit
- Aufgaben, Rollen und Verantwortlichkeit.

Bei den vorgenannten Anforderungen handelt es sich, mit Ausnahme der Interpretierbarkeit, um organisatorische Aspekte, deren Klärung zur Sicherstellung eines dauerhaften Betriebs erforderlich sind. Die Beschäftigung mit den Fragen wird auch begleitend zum Betrieb eines Archivs fortzusetzen sein.

Die Anforderungen an ein Langzeitarchiv werden vom Open Archival Information System (OAIS) Referenzmodell[4] (ISO-Norm 14721) identifiziert und in einen konzeptionellen Rahmen eingebettet, im nestor Handbuch[5] vom "nestor - Kompetenznetzwerk Langzeitarchivierung" werden die Anforderungen umfassen beschrieben.

Das **OAIS Referenzmodell** beschreibt Komponenten, Interaktionen und Funktionen eines Langzeitarchivs. Der Standard berücksichtigt Daten in Form von Informationspaketen, wobei unterschieden wird zwischen

- **Submission Information Package** (SIP), die Einlieferungseinheit, die beim Import (Ingest) an das Archiv übergeben wird,
- **Archival Informationen Package** (AIP), die im Archiv gespeicherte Archivierungseinheit, die alle zu einem Archivobjekt zugehörigen Dateien und auch (u.U. abgeleitete) beschreibende, technische und administrative Metadaten in Datenbanken umfasst, und
- **Dissemination Information Package** (DIP), die Exporteinheit, die bei Zugriff auf das Archiv wieder ausgeliefert wird, wobei der Umfang variieren kann, abhängig z.B. von rechtlichen Vorgaben oder gezielten Vereinbarungen für ein Archivobjekt, so dass das unveränderte SIP wieder ausgeliefert werden muss oder auch die Auslieferung eines abgeleiteten Pakets zulässig ist.

Das OAIS Referenzmodell macht keine Einschränkungen bzgl. der Nutzerdaten. Neben den Informationspaketen beschreibt der Standard ein Funktionsmodel, dass Aufgabenbereiche und Verantwortlichkeiten innerhalb eines digitalen Archivs festlegt: Datenübernahme (Ingest), Archivspeicher (Archival Storage), Datenverwaltung, Administration, Erhaltungsplanung (Preservation Planning) und Zugriff (Access), ohne Vorgaben für eine konkrete Implementierung oder Systemarchitektur zu machen. Das Funktionsmodell kennt die Rollen der Datenproduzenten, der Endnutzer und das Management. Die ersten beiden sind intuitiv verständlich. das Management legt Richtlinien für die Archivierung fest. Das OAIS Referenzmodell bildet einen Rahmen, der jedoch weiter konkretisiert werden muss.

2.5 Speicherungssysteme (Storage)

Das Archiv speichert die Daten und sichert deren Verfügbarkeit dauerhaft zu. Das heißt aber nicht, dass der Zugriff in jedem Fall unmittelbar erfolgen kann, weil die Datenspeicherung in einem Archiv, aus wirtschaftlichen Erwägungen, i.d.R. auf einem Bandspeicher erfolgt.

Um dennoch einen unmittelbaren Zugriff auf einen Teil der Daten zu ermöglichen, wird eine Trennung in Produktiv- und Archivspeicher vorgeschlagen. Beim Produktivspeicher (Online-Speicher) werden Daten auf schnellen aber teuren Platten- oder SSD-Speicher abgelegt, der einen direkten Zugriff auf die Daten erlaubt. Der Produktivspeicher kann Anfragen auch für ein Präsentationssystem angemessen schnell beantworten. Beim Archivspeicher (Offline-Speicher) handelt es sich um deutlich günstigeren Bandspeicher. Ein direkter Zugriff auf Daten

[4] http://www.oais.info

[5] URL für nestor Handbuch (Version 2.3): urn:nbn:de:0008-2010071949
http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:0008-2010071949

ist nicht unmittelbar möglich, weil der Leseauftrag in eine Auftragsliste eingereiht werden muss. Wenn der Auftrag an der Reihe ist, muss der Bandroboter zunächst das richtige Band holen und einlegen. Bevor dann gelesen werden kann, muss noch an die Speicherstelle vorgespult werden, das benötigt Zeit.

Um diese Trennung auf Datenebene zu realisieren ist es denkbar, dass

- durch eine Konfigurationsdatei festgelegt wird, welche Teile nicht im Produktivspeicher berücksichtigt werden sollen oder das
- anhand der Zugriffshäufigkeiten automatisch entschieden und zur Laufzeit in den Archivspeicher ausgelagert wird.

Im OLA-HD wird der erste Ansatz zur Trennung von Produktiv- und Archivspeicher verfolgt, um gezielt Bestandteile eines Archivobjektes für den Produktivspeicher auszublenden (s. 6 OLA-HD - Der Prototyp). Das in gepackter Form (als OCRD-ZIP) an das Archiv übergebene Importpaket (SIP, im Sinne des OAIS Referenzmodells), wird nach Übernahme im OLA-HD auf Speichersysteme und Datenbanken abgebildet. Damit wird das SIP in ein Archivabbild (das AIP) überführt. Das AIP umfasst

- das ZIP das unverändert im Archivspeicher abgelegt wird,
- die aus dem ZIP extrahierten Einzeldateien, die im Produktivspeicher für den schnellen Zugriff gespeichert werden, und ergänzend
- beschreibende, technische und organisatorische Metadaten, die in einer Datenbank abgelegt werden.

Der zweite Ansatz kann z.B. mit einem modernen Dateisystem vorgenommen werden, das eine Komponente zum **hierarchischen Speichermanagement (HSM)**[6] umfasst. Dateisysteme die HSM unterstützen (z.B. StorNext File System[7] oder OpenArchive[8]) können so konfiguriert werden, dass Daten (d.h. Dateien), auf die längere Zeit nicht zugegriffen wurde, in einen Auslagerungsbereich im Dateisystem verschoben werden, von wo sie dann bei Bedarf, d.h. um Platz zu schaffen, auf das Band ausgelagert werden. Es bleibt lediglich eine Markierung im Dateisystem bestehen. Bei Zugriff veranlasst das Dateisystem eine Wiedereinlagerung. Die Aus- und Einlagerung von einzelnen Dateien ist jedoch nicht performant und kann dazu führen, das zusammengehörende Dateien auf unterschiedliche Bänder ausgelagert werden, was dazu führt, dass der Bandroboter nur noch mit Bandeinlegen/-auswerfen und Spulen beschäftigt ist und den Archivspeicher lahmlegt. Deshalb werden nicht Einzeldateien, sondern ganze Bereiche des Dateisystems in gepackter Form ausgelagert.

2.6 Archiv-Manager CDSTAR

Die Datenverwaltung und die Trennung von Produktiv- und Archivspeicher, und eine prüfsummen-basierte Adressierung von Dateien wird im OLA-HD über den Archiv-Manager

[6] Eng. Hierarchical Storage Management (HSM)

[7] https://www.quantum.com/en/products/file-system/ (kommerziell)

[8] https://www.graudata.com/openarchive (open source)

GWDG Common Data Storage Architecture (CDSTAR)[9] gesteuert. Die Aufgaben die der Archiv-Manager zu erfüllen hat umfassen:

- Speicherung des Importpaketes im Archivspeicher (mit eigener PID)
- Entpacken des Importpakets und Speicherung ausgewählter Dateien im Produktivspeicher, die gemäß Konfiguration für den direkten Zugriff vorgesehen sind (mit eigener PID)
- Speicherung von Informationen zu Import und Archivobjekt in einer Datenbank (Wann und von Wem durchgeführt, Umfang, Ablaufdatum für Produktivspeicher, etc.)
- Ermittlung und Speicherung der Werte des Basismetadatensatz (s. 4.4 Metadaten) in Suchindex
- Volltextindexierung text-basierter Importdaten
- Bereitstellung der Archivdaten für den Export
- Ein-/Auslagerung von Daten zur Speicherplatzoptimierung

Da die Verarbeitungsschritte beim Import einige Zeit in Anspruch nehmen, laufen sie für den Benutzer im OLA-HD im Hintergrund. Der Benutzer muss den Status des Importauftrags über die Benutzungsschnittstelle überprüfen.

Der Archiv-Manager entscheidet anhand einer Konfiguration, welche Teile der Importdaten im Produktivspeicher abgelegt werden. So wird bspw. festgelegt, das hochauflösende Bilder (TIFFs) nicht in den Produktivspeicher kopiert werden, womit sich das dort zu speichernde Datenvolumen deutlich reduziert. Als weiteren Schritt der Optimierung wird festgelegt, dass Daten aus dem Produktivspeicher gelöscht werden, wenn länger als 3 Wochen nicht auf sie zugegriffen wurde. Die Wiedereinlagerung erfolgt transparent, kann jedoch einige Zeit in Anspruch nehmen. Um die Belegung des Produktivspeichers noch weiter zu optimieren, werden Daten nicht unter Ihrem Dateinamen gespeichert, stattdessen wird eine Datei nach der Prüfsumme der Datei benannt und unter diesem Namen gespeichert. Damit führen Duplikate nicht zu einer Mehrfachspeicherung (Dateibasierte Deduplizierung). Bei Versionen wird dadurch ein großes Einsparpotential erzielt.

2.7 REST API[10]

Für die Maschine-zu-Maschine Interaktion mit dem OLA-HD wurde sich für einen **Representational State Transfer (**REST)[11] basierten Web Service Ansatz entschieden. REST ist gut dokumentiert, weit verbreitet und einfach einzusetzen, und durch strikte Nutzung der HTTP Methoden quasi standardisiert, was auch die Begründung für die Auswahl ist. Beim REST Ansatz hat das Archiv zudem keine Kenntnis von den Datenlieferanten, es besteht insofern keine direkte Abhängig zu externen Systemen.

Bei einer REST-basierten Programmierschnittstelle (Application Programming Interface, bzw. API) können mit Kenntnis des Service Endpunktes (per URL) und dem Namen einer Ressource

[9] https://cdstar.gwdg.de, API: https://www.gwdg.de/documents/20182/31188/gwdg-bericht-78.pdf

[10] Programmierschnittstelle, eng. Application Programming Interface (API)

[11] Dissertation von Roy Fielding, die den Begriff REST maßgeblich geprägt hat, https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf

alle CRUD Funktionen aufgerufen werden, indem ein HTTP Aufruf abgesendet wird, der die jeweilige HTTP Methoden (POST, GET, PUT, DELETE) benutzt. Bei einer POST Anfrage wird bspw. ein neues (Archiv)Objekt erzeugt, mit GET wird gelesen, PUT wird für Änderungen verwendet und DELETE führt zum Löschen. Eine REST Anfrage zum Lesen eines Objekts kann auch aus dem Browser gesendet werden, da der Browser eine HTTP GET Anfrage absendet. Das erzeugen einer neuen Ressource über eine POST Anfrage erfolgt z.B. über eine Bibliotheksfunktion einer HTTP-Software-Bibliothek.

3 Anforderungen

Die Kernanforderungen werden durch die Ziele des Projekts OCR-D und des Moduls, wie sie in der Einleitung eingeführt wurden, bestimmt. Die Anforderungen wurden auf Basis des Projektantrages konkretisiert und in Absprache mit den OCR-D Projektpartnern in der laufenden Entwicklung angepasst und weiterentwickelt.

Im nachfolgenden werden die Anforderungen an ein Archiv beschrieben. Es wird zwischen strengen Kernanforderungen, die quasi verpflichtenden Charakter haben, und ohne deren Erfüllung der Nutzen des Archivs in Frage gestellt ist und solche, die sich die Anwender ergänzend wünschen, unterschieden. Die Funktionen der ergänzenden Anforderungen könnten auch durch externe Dienste erfüllt werden.

3.1 Anwendergruppen

Die Anforderungen an das Archiv werden bestimmt durch Anwender, die mit dem Archiv arbeiten. In OCR-D wurden folgende drei Anwendergruppen identifiziert, die Anforderungen an das Archiv stellen:

- **Datenproduzenten**, z.B. Institutionen und wissenschaftliche Projekte mit Digitalisierungs- und Erschließungsaufgaben
- **Betreiber von Repositorien und Portalen,** z.B. Archive, Bibliotheken, Museen, Universitäre Einrichtungen, und andere Gedächtniseinrichtungen
- **Anwender**, z.B. wissenschaftliche Anwender und Projekte, die OCR-Daten verwenden möchten

Für OCR-D steht zunächst die Gruppe der Datenproduzenten[12] im Vordergrund, da es die intendierte Anwendergruppe ist. Diese möchten die Versionen ihrer OCR-Daten langfristig verfügbar (i.S.v. benutzbar und durchsuchbar) halten. Für die zweite Gruppe, die aus funktionaler Sicht, dieselben Anforderungen an das System stellt, kommen Aspekte hinzu, die bspw. den Bestandserhalt oder die Erfüllung rechtlicher oder formaler Vorgaben, und Maßnahmen für die LZA (z.B. Speichermigrationen und Formattransformationen) betreffen, zu denen sie Richtlinien ausarbeitet, und sie sichert die wirtschaftlichen und organisatorischen Rahmenbedingungen für den Betrieb. Anwender der dritten Gruppe möchten mit den Daten

[12] Die Trennung der Gruppen Datenproduzenten und Betreiber von Repositorien ist nicht zwingend, da sie funktional kongruent sind. Die Trennung dient hier der Hervorhebung.

arbeiten und z.B. Text und Data Mining Prozesse durchführen, um den Datenbestand zu analysieren. Für sie ist die dauerhafte Referenzierbarkeit, Zitierbarkeit und Verfügbarkeit der referenzierten Objekte ein Qualitätskriterium, weil ihre Arbeit nur damit überprüfbar ist.

3.2 Kernanforderungen

3.2.1 Authentizität

Mit Authentizität wird die Erwartung umschrieben, dass Daten unverfälscht und vollständig sind. Es muss insofern sichergestellt werden, dass der Datentransfer zum und die Datenübernahme am Archiv korrekt verlaufen ist, dass die Daten im Archiv unverändert gespeichert und auch wieder bereitgestellt werden können und das insbesondere weder beim Datentransfer, noch im Archiv unerwünscht Veränderungen vorgenommen werden. Die Authentizität betrachtet alle OAIS-Informationspakete, also den Datenimport (SIP), die Speicherung im Archiv (AIP), als auch den Datenexport (DIP). Um dies sicherzustellen zu können, sind Mechanismen erforderlich, die eine Prüfung der Datenübertragung und der gespeicherten Archivobjekte ermöglichen. Das kann bspw. auf Basis von Prüfsummen erfolgen, die bei der Datenübergabe mitgeschickt werden und die auch im Archiv zu speichern sind. Im Datenabschnitt wird das Datenaustauschformat OCRD-ZIP vorgestellt (s. 4.3.1 OCRD-ZIP), das Prüfsummen zu den Dateien des Importpaketes beinhaltet, mit denen die Authentizität geprüft werden kann.

Zur Authentizität gehört auch die Möglichkeit die Entstehungshistorie eines Archivobjekts abbilden zu können. Damit kann nachverfolgt werden, wie der aktuelle Zustand oder die aktuelle Version des Archivobjekts durch erwünschte Änderungen, z.B. durch für LZA erforderliche Speichermigrationen und Formattransformationen, zustande gekommen ist.

3.2.2 Integrität

Die Datenintegrität zielt auf Mechanismen ab, die sicherstellen, dass Daten technisch Korrekt sind. Redundante Speicherung von Daten oder ein Backup ermöglichen die Datenwiederherstellung bei Datenverlust. Um Datenverlust zu verhindern, müssen regelmäßige Überprüfungen der Datenträger und Migration der Daten auf neue Datenträger vorgenommen werden. Die Integrität wird bis auf die Speicherebene, d.h. die Abfolge der gespeicherten Bits (Bitstream) einer Datei im Dateisystem heruntergebrochen und es ist erforderlich, dass diese Abfolge über die Zeit und auch nach Migrationen korrekt bleibt (Bitstream Preservation). Nur so bleiben Dateien lesbar und interpretierbar.

Eine weitere Form der Datenmigration ist die Formatmigration zum Zweck der LZA (s. 4.1 Vorbemerkung), zur Sicherstellung der Interpretierbarkeit der Daten. Nach einer Formatmigration ist die Integrität zwangsläufig nicht mehr gegeben. Nur durch strukturierte Dokumentation der Änderungen kann eine "relative Integrität" geschaffen werden. Mit dem Metadaten Standards ProvOne (s. 4.5.4 ProvOne) und PREMIS (s. 4.5.5 PREMIS) kann der Nachweis geführt werden, wer, welche Veränderungen, wann, womit durchgeführt hat und zu welchem Ergebnis diese Änderung geführt hat.

Die Integrität kann aber auch weiter gefasst werden und auch die Eindeutigkeit und Widerspruchsfreiheit umfassen. Der Import einer neuen Version eines Archivobjekts darf

keine Auswirkungen auf andere Versionen haben und eine persistente ID (PID) muss für alle Zukunft genau eine bestimmte Version eines Archivobjekts identifizieren, was wiederum bedingt, das PIDs nicht wiederverwendet werden dürfen.

Auf technischer Ebene wird die Datenintegrität des OLA-HD durch die technische Infrastruktur, sichergestellt, in der das Archiv betrieben wird. Daten werden per Backup gesichert und das eingesetzte Dateisystem sollte die Aspekte der Bitstream Preservation beachten, d.h. es speichert Daten redundant, um Konsistenzchecks und die Wiederherstellung defekter Daten zu ermöglichen.

Das OLA-HD integriert den Handle.Net[13] PID Dienst und garantiert, dass eine spezifische Version eines Archivobjekts über eine PID dauerhaft und eindeutig adressiert wird. Die Importschnittstelle und der Archiv-Manager (CDSTAR) stellen sicher, dass für jeden Import ein neues Archivobjekt erzeugt wird. Bei Import mit Nennung einer Vorgängerversion wird in den Metadaten eine Beziehung zwischen den beiden Archivobjekten gesetzt. Importe sind damit unabhängig voneinander und können sich nicht überschreiben oder beeinflussen, abgesehen von dem Setzen der Beziehung.

3.2.3 Verfügbarkeit

Die Erwartung der verschiedenen Nutzergruppen ist, dass das Archiv für den Zugriff erreichbar ist, sowohl für den Datenimport, als auch den Datenzugriff. Der Zugriff auf die Daten soll in angemessener Zeit möglich sein. Bei der Anfrage an ein Archiv darf die Antwortzeit eine größere Zeitspanne in Anspruch nehmen, als bspw. bei einem institutionellen Repositorium, bei dem Antwortzeiten i.a. im Millisekunden Bereich liegen sollten. Es wird erwartet, dass die Daten dauerhaft und stabil verfügbar sind und nicht verlorengehen. Für die Verfügbarkeit der Daten ist deren Integrität eine integrale Voraussetzung.

Abhängig vom PID Konzept und den formalen Bestimmungen des PID Service (s. 3.2.6 Identifikation von Archivobjekten) verpflichtet sich der Servicenutzer, dass die von der PID identifizierte und über eine damit erzeugte PURL referenziert Ressource dauerhaft verfügbar ist. Wenn das Archivobjekt gelöscht oder in irgendeiner Form verborgen wird, dann entsteht ein Bruch. Zitationen und Referenzen wären nicht mehr gültig. Es sind jedoch Gründe denkbar, die eine weitere Bereitstellung bzw. den weiteren Zugriff verbieten (z.B. Nutzungsrechte oder verwerflicher Inhalt). In solchen Fällen muss im Einzelfall geprüft werden, ob das Archivobjekt gelöscht (oder als gelöscht markiert) werden muss. Um die gegebene Garantie einzuhalten, sollte eine PID auch weiterhin aufgelöst werden und nicht ins Leere zielen. Bei Zugriff sollten mindestens die Metadaten des Archivobjekts geliefert werden, ggf. ergänzt um Informationen, warum das Objekt nicht mehr erreichbar ist und um Kontaktdaten, so dass eine Nachfrage im Bedarfsfall möglich ist. Das kann sowohl über den PID Resolver Dienst (vgl. 3.2.6 Identifikation von Archivobjekten) implementiert werden, als auch auf Applikationsebene. Eine derart weitreichende Funktion wie das Löschen oder Verbergen sollte in gut begründeten Ausnahmefällen ausschließlich durch einen Administrator durchzuführen werden können.

Für den Zugriff und die Integration von Archivdaten sind Schnittstellen für menschliche Nutzer und auch für software-technische Nutzer (z.B. Dienste und Prozesse) erforderlich. Gerade für

[13] https://hdl.handle.net/

die Analyse von Datenbeständen (z.B. Text and Data Mining) muss die Schnittstelle Anfragen nach großen Datenmengen stabil bedienen können, ohne dass dadurch die Verfügbarkeit des Archivs, bspw. durch Überforderung der Schnittstelle oder Systemabsturz, gefährdet wird.

Die Schnittstelle des OLA-HD ist REST-basiert und das Archiv kann Anfragen (Import, Zugriff, Suche) zeitgleich verarbeiten. Die aktuelle OLA-HD Installation besteht aus einer einzelnen Instanz. Sollte die Konfiguration aus Gründen der Anfragelast an seine Grenzen stoßen, so kann die Verfügbarkeit durch verteilte Installation mehrerer Instanzen (Clusterisierung) verbessert werden.

3.2.4 Vertraulichkeit

Die Zugriffsberechtigung auf Archivobjekte kann durch Lizenzen und Nutzungsrechte eingeschränkt werden. Um vor unberechtigten Zugriff zu schützen, müssen Mechanismen existieren, die einen autorisierten Zugriff kontrollieren, so dass der Zugriff bspw. gruppen- oder rollenbasiert erlaubt werden kann. Das erfordert ein Berechtigungskonzept, das Nutzer-, Gruppen- und Rollen beinhaltet, und es ermöglicht, Archivobjekte mit Zugriffsrechten zu belegen.

Das Nutzer- und Berechtigungsmanagement wird aktuell durch einen Verzeichnisdienst (Active Directory) erfüllt. Für das OLA-HD ist der Datenproduzent/-lieferant der Eigentümer des Archivobjekts. In der aktuellen Installation des OLA-HD ist es noch nicht möglich Berechtigungen auf Objekten festzulegen, und Zugriffe entsprechend zu kontrollieren. Zugriffsberechtigungen müssen bei großen Datenbeständen automatisiert aus den Metadaten der Importpakete abgeleitet werden. Da das OLA-HD aktuell ausschließlich gemeinfreie Daten speichert (VD18), ist das Fehlen der Funktion jedoch unproblematisch.

3.2.5 Sicherheit

Neben den vorgenannten funktionalen Anforderungen, die alle einen spezifischen Aspekt der Datensicherheit berücksichtigen, muss das Archiv selbst vor unberechtigtem Zugriff geschützt werden. So muss das Berechtigungskonzept bspw. Schutz vor unberechtigter Datenübergabe bieten. Im OLA-HD muss die Schreibberechtigung für Nutzer explizit gesetzt werden.

Im Projekt ist es möglich, ein GWDG Nutzerkonto über die GWDG Portalseite (GWDG Nutzerkonto erstellen[14]) zu beantragen. In einem zweiten Schritt wendet sich der Nutzer an den GWDG Support [15], mit der Bitte, das neue Konto für den OCRD/OLA-HD Kontext freizuschalten. Der Nutzer wird dann in eine entsprechende Active Directory Gruppe aufgenommen.

3.2.6 Identifikation von Archivobjekten

Die Anforderung nach dauerhafter Adressierbarkeit und Zitierbarkeit von Archivobjekt Versionen macht es erforderlich, dass diese eindeutig und dauerhaft, d.h. persistent identifiziert werden können. Dies kann auf unterschiedlichen Ebenen stattfinden. So kann eine Version des Archivobjekts als Ganzes, d.h. als Einheit, identifiziert werden oder es ist eine

[14] https://www.gwdg.de/registration

[15] GWDG Support (support@gwdg.de)

Identifizierung der Fragmente (s. 3.3.1 Identifizierung von Fragmenten und Duplikaterkennung) unterhalb von bibliographischen und physischen Einheiten (z.B. Kapitel, Artikel, Seiten, Seitensegmente, etc.), möglich. Es ist natürlich auch beides zugleich möglich. Ein Archiv sollte mindestens die Identifizierung und damit die Adressierung des Archivobjekts als Ganzes (AIP) unterstützten.

Die Funktion der persistenten Adressierung wird erreicht, indem eine PID auf eine **persistente URL (PURL)**, also einen persistenten Verweis, abgebildet wird. Beim Handle Service hat die PURL die Form http://hdl.handle.net/21.T11998/0000-001A-69F3-C, wobei es sich bei der Endung 21.T11998/0000-001A-69F3-C um die PID handelt. Ein externer Dienst, der **PID Resolver Service**, kann die PURL auflösen und auf eine konkrete URL abbilden. Die Zwischenkomponente des Resolvers bringt den Vorteil mit sich, dass die PURL auch dann gültig gehalten und aufgelöst werden kann, wenn die URL der Archivschnittstelle sich geändert hat. Um das zu gewährleisten, muss das Mapping im Resolver angepasst werden. Die Adresse des Resolver muss stabil bleiben.

Im OLA-HD wird nach der Übernahme des Importpakets für jedes Archivobjekt vom **PID Service** (GWDG PID Service[16]) eine neue **PID** (Handle ID) bezogen und registriert. Sowohl das Objekt im Produktivspeicher, als auch das im Archivspeicher bekommt eine eigene PID. Für die Auflösung der PIDs wird ebenfalls der Handle.Net Dienst eingesetzt, der eine Weiterleitung auf das Archivobjekt vornimmt. Die PID wird auch in den Metadaten des Archivobjekts als Identifier aufgeführt. Der Aufruf der PURL im Browser führt zum Download des referenzierten Archivobjekts.

3.2.7 Versionierung

Ein Archiv muss nicht zwangsläufig die Versionierung von Archivobjekten unterstützen. Für das OLA-HD ist es jedoch eine Kernanforderung, da die Speicherung und der Vergleich von Versionen und insbesondere ihrer Prozessierungskonfigurationen im Kontext von OCR-D im Vordergrund stehen. Die Erwartung ist insofern, dass alle Versionen archiviert werden und auf frühere Versionen ebenso komfortabel zugegriffen werden kann, wie auf die zuletzt importierte, d.h. aktuelle, Version. Dazu ist es erforderlich, dass jede Version eine eigene eindeutige PID zur Identifizierung zugeordnet bekommt.

Bei Mehrfachimport des gleichen Importobjekts sind verschiedene Szenarien zu beachten. So kann ein Import mit Angabe der PID einer **Vorgängerversion** stattfinden und damit klar die Beziehung zu einem bestehenden Archivobjekt kennzeichnen und eine Version erzeugen. Fehlt dieser Hinweis, so erzeugt jeder Import zunächst ein eigenständiges Archivobjekt. Es kann beim Import jedoch auch geprüft werden, ob es bereits ein Archivobjekt mit demselben Identifier gibt, so kann auch festgelegt werden, dass der neue Import eine Version zum bestehenden Archivobjekt erzeugen soll. Das ist jedoch in zweierlei Hinsicht problematisch:

1. Damit wird unterstellt, dass ein Import immer eine Version zu einem bestehenden Archivobjekt erzeugen soll. Ggf. ist es aber gut begründet, dass es mehrere Archivobjekte zur selben Vorlage geben soll, die eigenständige und nicht zueinander in Beziehung stehen sollen.

[16] https://www.gwdg.de/application-services/persistent-identifier-pid

2. Da der Einordnungspunkt nicht bekannt ist, würde ein Import quasi immer hinten angehängt, d.h. eine neue letzte Version gebildet.

Ein konkretes Beispiel für Fall 1. wäre, dass zwei Institutionen dasselbe Archivobjekt prozessieren, und erwarten, dass sich ihr Import in einer eigenen Versionshistorie wiederfindet, d.h. ein eigenständiges Archivobjekt bildet.

Bezüglich Fall 2. ist vorstellbar, dass ein initialer Import bspw. nur Bilddaten und Ground Truth umfasst, also eine Basis für die nachfolgenden Prozessierungen bildet. Wenn in der Folge Prozessierungen mit verschiedenen Konfigurationen durchgeführt werden, dann sollen diese nicht hinten angehängt, sondern nebeneinander in der Objekthistorie stehen und eigene Zweige (Branches) bilden. Die (direkt) nachfolgenden Versionen referenzieren alle dasselbe Basis-Archivobjekt als Vorgänger.

Die Referenzierung kann über die Metadaten erfolgen, indem die neue Version die PID der Vorgängerversionen speichert. Da Versionen im Laufe der Zeit entstehen, haben Version keine Kenntnis über ihre Nachfolgeversionen. Die Information kann nachgetragen werden, was jedoch den Zustand der Vorgängerversion ändert, obwohl sich diese eigentlich nicht geändert hat.

Es wurde daher die **Festlegung für OCR-D** getroffen und so im OLA-HD implementiert, dass

- jeder Import ein eigenständiges Archivobjekt mit eigener PID erzeugt,
- nicht automatische abgeleitet wird, ob es sich um eine neue Version eines bestehenden Archivobjekts handelt, wenn der Import ohne Nennung der Vorgängerversion erfolgt, und dass
- Importe mit Nennung der Vorgängerversion ggf. eine hierarchische Versionshistorie erzeugen, wobei Zweige dadurch entstehen, dass eine spezielle Version Vorgänger verschiedener Folgeversionen ist.

Im OLA-HD wird die Vorgängerversion beim API Aufruf als Parameter übergeben und in der Datenbank als Metadatum gespeichert.

3.2.7.1 Qualitätsverbesserung durch Versionen

Im Projekt OCR-D steht die Verbesserung von OCR-Prozessen und -Workflows im Vordergrund. Für die Qualitätsverbesserung ist die wiederholte Prozessierung derselben Vorlage mit geänderter OCR-Konfigurationen (OCR Engine, Algorithmus, Trainingsprozesse- und status, technische Umgebung, etc.) und der Vergleich der Konfigurationen auf Basis der Prozessierungsergebnisse erforderlich. Die Vergleichsergebnisse werden zur Beurteilung der Prozessierungskonfiguration herangezogen. Mit dem neu gewonnenen Wissen kann anschließend eine neue Prozessierung mit angepassten Konfigurationen durchgeführt werden. Es entstehen dabei Versionen. Mit jeder Version wird der Pool an dokumentierten Prozessierungen vergrößert. Dieser Prozess wird in mehreren Iterationen wiederholt, um herauszufinden, welche Änderungen sich Qualitätsverbessernd oder -verschlechternd auswirken, um Konfigurationen sukzessive zu verbessern, und um beurteilen zu können, welche Konfiguration für diesen oder jenen Einsatzbereich optimal ist, um davon ausgehend

eine Prozesskonfiguration, bspw. für bestimmte Korpora und Schrifttypen wie Antiqua, Fraktur und Handschrift, auswählen zu können

Um eine Beurteilung der OCR-Konfigurationen vornehmen zu können, ist es erforderlich, dass jede Version neben den Eingabedaten (Bilder, beschreibende Metadaten, Ground Truth), den Ergebnisdokumenten (OCR Ergebnisse mit Fehler- bzw. Erkennungsrate,), auch Provenienzdaten, d.h. eine Beschreibung der Prozesse (Objekte, Aktivitäten bzw. Ereignisse, Akteure und Rechte), die eine Zustandsänderung herbeigeführt haben, gespeichert werden. Ein geeignetes Metadatenformat zur Dokumentation der Provenienzdaten ist bspw. ProvOne (s. 4.5.4 ProvOne). Mit den Metadaten der Prozessierung kann die Entstehung von Zuständen nachvollzogen werden, bei ausreichendem Detaillierungsgrad ist sogar eine Reproduktion der Zustände möglich. Informationen können indexiert werden, so dass gefilterte oder komplexe Suchen unterstützt werden, um bspw. OCR-Ergebnisse zu finden, deren Vorlage einen konkreten Schrifttypen haben, mit einer bestimmten OCR Engine prozessiert wurden und zu einer bestimmten Qualität (Wortfehlerrate oder Zeichenfehlerrate) geführt haben.

Es wird deutlich, dass die Position der Version in der Versionshistorie von nachrangiger Bedeutung ist. Entscheidend ist das Ergebnis und wie es zustande gekommen ist.

3.2.8 Schnittstellen

Die Interaktion mit dem Archiv sollte über eine schlanke und klar definierte Schnittstelle stattfinden. Das reduziert und vereinfacht Abhängigkeiten und die Komplexität der Gesamtarchitektur.

Jenseits der Abhängigkeiten, die durch Schnittstelle entstehen, gibt es Abhängigkeiten durch die Festlegung auf Austausch- und Beschreibungsformate. Wie im Datenabschnitt (s. 4 Daten) beschrieben, ist es nicht möglich ein generisches Archiv für beliebige Datenformate zu entwickeln, da dadurch ein sehr großer Komplexitätsgrad entsteht, der die (Weiter-)Entwicklung und den stabilen Betrieb sehr aufwändig macht. Das Internet Archive (s. 5.1.8 Internet Archive) bspw. unterstützt relativ viele Dokumenttypen (Bücher, Videos, Audio, etc.). Um dies zu ermöglichen, ist die Schnittstelle, auf einen kleinen Satz an Metadaten und eine ganz konkrete Struktur der Importdaten eingeschränkt. Ein Import eines Buchs bspw. kann im ZIP Dateiformat stattfinden, wobei alle Bilder des Buches im Paket enthalten sein müssen. Es können aber keine weiteren Daten wie Volltexte, bibliographische Metadaten, Prozessierungsinformationen, etc. importiert werden. Die Archivierung der OCR-D Daten wäre damit nicht möglich.

Die Schnittstelle muss die Integration auf unterschiedlichen Ebenen unterstützen, Daten sollen direkt aus Digitalisierungs- oder OCR-Prozessierungsworkflows ins Archiv übertragen werden, automatisierte Prozessierungen sollen Daten aus dem Archiv laden können, um Textanalysen wie z.B. Named Entity Recognition (NER) durchzuführen, oder bestehende Datenbestände sollen nachträglich ins Archiv überführt werden. Die Implementierung der Schnittstelle über eine REST API (s. 2.7 REST API) bildet eine gute Basis für die Integration.

Im Folgenden werden Schnittstellen beschrieben, die vom OLA-HD angeboten werden.

3.2.8.1 Benutzungsschnittstelle

Für eine direkte Interaktion mit menschlichen Nutzern, sollte das Archiv eine Benutzungsschnittstelle bieten, die eine einfache Form der Suche, Präsentation und den Export von Archivobjekten unterstützt.

Das OLA-HD beinhaltet eine Benutzungsschnittstelle, über die der Nutzer das Archiv durchsuchen und einen Download veranlassen kann. In einer Detailansicht werden die Metadaten angezeigt. Über eine Auswahlkomponente ist ein direkter Zugriff auf Archivobjekte möglich. Einzelne Dateien können darüber ausgewählt und im Browser angezeigt werden, um z.B. den Text einer Seite zu lesen oder vor einem Download beurteilen zu können, ob es sich bei dem ausgewählten Archivobjekt um das richtige handelt. Zudem können über die Auswahlkomponente einzelne Dateien oder auch Dateigruppen zum Download markiert werden, um sie als ZIP Archiv zu exportieren.

Für Benutzer die sich am System angemeldet haben bietet die OLA-HD Benutzungsschnittstelle eine Übersichtsseite, auf welcher der Status laufender Importprozesse angezeigt wird. Es wäre auch möglich dem Datenlieferanten den Status per E-Mail mitzuteilen, was aber bei großen Importvorhaben sehr viele Benachrichtigungen nach sich zieht, so das sich gegen diese Option entschieden wurde. Importaufträge können sich im Status processing, success oder failed befinden. Den Status zu prüfen und entsprechend zu reagieren obliegt dem Nutzer oder dem Dienst der den Import angestoßen hat.

3.2.8.2 Import API

Der Import kann gebündelt, in gepackter Form stattfinden, oder durch Einzelschritte. Im zweiten Fall ist jedoch nicht klar, ab wann ein Archivobjekt vollständig importiert ist und welcher Zustand eine vollständige Version beschreibt. Der Zustand des Archivs zu einem bestimmten Zeitpunkt müsste explizit als Version markiert werden. Das kann z.B. wie in GitHub (s. 5.1.7 GitHub) geschehen, wobei der Zustand des Archivs zu einem bestimmten Zeitpunkt explizit durch setzen einer Marke (Tags) als Version markiert wird. Beim Fedora Commons Repository (s. 5.1.6 Fedora Commons), bildet jede Änderung einer Einzeldatei eine Version der Datei. Beides ist für OCR-D ungünstig, da das Setzen einer Marke vergessen werden kann und beim Versionieren auf Dateiebene das Archivobjekt als Ganzes keine Version hat.

Ein Archivobjekt entsteht nicht inkrementell, sondern stellt einen stabilen Zustand dar, der in OCR-D das Ergebnis eines Prozessierungsdurchlaufs ist. Wenn der Zustand nicht stabil ist, dann gehört das Objekt noch nicht ins Archiv. Insofern ist die Festlegung in OCR-D auf das Importformat OCRD-ZIP (s. 4.3 Austauschformat) konsequent und damit zudem OAIS konform. Ein Importpaket umfasst damit immer eine vollständige Version, mit allen zugehörenden Daten und Metadaten.

Importdaten müssen nach der Datenübernahme auf Korrektheit, sowie auf Konformität zum geforderten Importformat überprüft werden. Damit die Importschnittstelle nicht blockiert und quasi zeitgleich viele Importaufträge entgegennehmen kann, muss sie die Importaufträge in eine Warteschlange (Queue) einreihen, die dann von parallel arbeitenden Worker Instanzen abgearbeitet werden.

Das OLA-HD nimmt Pakete im OCRD-ZIP Paketformat entgegen und validiert den Import auf Korrektheit, Vollständigkeit und Konformität. Danach kann die korrekte Datenübernahme bestätigen werden oder es wird eine Fehlermeldung zurückgemeldet. Bei korrekter Übernahme wird der Archiv-Manager für das Importpaket PIDs erzeugen, die Speicherung in Produktiv- und Archivspeicher veranlassen, Datenbankeinträge vornehmen, sowie Metadaten und Volltexte indexieren. Die Datenübernahme erfolgt beim OLA-HD über eine REST-basierte API (s. 6.5 API Beschreibung). Der API Aufruf kann mit Nennung der Vorgängerversion als Parameter erfolgen, womit der Import eindeutig als Version eines bereits existierenden Archivobjekts markiert wird. Die Schritte werden bei der Beschreibung des OLA-HD noch detaillierter ausgeführt (s. 6.4 Interaktion mit dem Archiv).

Alternativ zu einem nutzergesteuerten Import, wäre auch ein Harvestingansatz denkbar, wie er bspw. im Abschnitt 3.3.3 Einsammeln der Daten von verschiedenen Orten skizziert wird. Dabei könnte die METS Datei von einem Harvesting Server bereitgestellt werden oder vom Datenlieferanten aktiv über die Importschnittstelle an das Archiv übergeben werden. Die Daten müssen von den im METS referenzierten Standorten geladen werden. Beim OLA-HD wurde sich jedoch gegen den Harvesting Ansatz entschieden, weil nicht davon ausgegangen werden kann, dass Datenlieferanten über eine entsprechende Schnittstelle verfügen, und selbst wenn die Schnittstelle existiert, wird damit eine nicht kontrollierbare Abhängigkeit zu einem externen System eingegangen, die viel Wartungs- und Pflegeaufwand verursachen kann. Erschwerend kommt bei diesem Ansatz hinzu, dass die Daten archiv-seitig gepackt werden müssen, um sie im Archivspeicher abzulegen.

3.2.8.3 Export API

Das OLA-HD bietet für den Datenexport drei Optionen:

1. Vollständiger Export aus dem Archivspeicher per API
2. Vollständiger Export des Ausschnitts im Produktivspeicher per API
3. Export von Bestandteilen aus dem Produktivspeicher über die Benutzungsschnittstelle

Für 1. und 2. wird der Export im OCRD-ZIP Paketformat vorgenommen (s. 4.3 Austauschformat). Der Zugriff auf das Archivobjekt erfolgt entweder mit der PURL (s. 3.2.6 Identifikation von Archivobjekten), oder dem von der OCRD-METS Konvention geforderten eindeutigen Identifier (s. 4.5.1 METS). Der Export in 3. erfolgt als ZIP, entsprechend der Ablagestruktur im Archiv, nicht aber als OCRD-ZIP Paket.

3.2.8.4 Update API

Die Erwartung bei Archivobjekten ist, dass diese einen stabilen Zustand haben und das auf ihnen nicht mehr aktiv gearbeitet wird. Daher erzeugt ein Import immer ein neues Archivobjekt oder eine Version zu einem bestehenden Archivobjekt. Ein Reimport ist insofern nur ein erneuter Import, Updates gibt es insofern nicht.

Es soll dennoch skizziert werden, was es bedeutet, wenn Updates zulässig wären, um zu verstehen, dass es konsequent ist, Updates zu verbieten:

Ein Update wäre bei einer Konfiguration, wie sie beim OLA-HD vorliegt, ausgesprochen kompliziert. Eine Update Schnittstelle könnte eine einzelne Datei entgegennehmen oder ein

komplettes Archivobjekt. Bei der Einzeldatei müsste diese im Produktiv- und Archivspeicher aktualisiert werden. Das bedeutet, dass das Archivobjekt vom Band gelesen werden muss, die geänderte Datei müsste im ZIP aktualisiert und das ZIP dann wieder auf das Band geschrieben werden. Danach muss die alte Version vom Band gelöscht werden. Allein die Anzahl der notwendigen Aktionen und Abhängigkeiten macht das System fehleranfällig. Wenn es nicht möglich ist, den Workflow als geschlossene Transaktion zu implementieren, die im Fehlerfall zurückgenommen werden kann, dann birgt schon das Update einer Einzeldatei erhebliche Risiken. Bei Übergabe und Update eines kompletten Archivobjekts, wird die Aufgabe noch anspruchsvoller. Neben dem Update der einzelnen Dateien muss eine Entscheidung getroffen werden, was mit Dateien geschehen soll, die auf dem Band existieren, aber nicht Teil des Updatepaketes sind. Es muss entschieden werden, ob diese gelöscht oder weiter existieren sollen.

3.2.8.5 Delete API

Im Abschnitt 3.2.2 Integrität wurde bereits darauf eingegangen, dass das Löschen von Archivobjekten problematisch ist. Das OLA-HD unterstützt nicht das Löschen von Archivobjekten per API. Wenn das Löschen oder deaktivieren eines Archivobjekts aus wichtigem Grund durchgeführt werden muss, dann muss sich der Benutzer an einen Administrator wenden, der die Notwendigkeit überprüft und das Löschen oder Verbergen ggf. durchführt.

3.2.8.6 Suche über Metadaten

Es Archiv muss mindestens über eine basale Suchschnittstelle verfügen, um Archivobjekte und ihre Versionen wiederfinden zu können. Da ein Archiv nicht als Such- oder Recherchesystem konzipiert ist, wird der Funktionsumfang bzgl. der Suche eher einfach ausfallen. Es soll mindestens eine einfache Metadatensuche möglich sein (z.B. nach Titel oder Autor), ggf. ergänzt um Filtermöglichkeiten. Die Suche kann zudem ergänzt werden durch eine Volltextsuche, bei der alle Archivobjekte analysiert und indexiert werden. Die Volltextsuche kann jedoch bei einem großen Datenbestand und unspezifische Suchtermen, ohne zusätzliche Filterungsmöglichkeiten, zu großen und schwer nutzbaren Treffermengen führen.

Die Auswahl der "richtigen" Metadaten ist nicht einfach. So wäre die Auswahl bei Zeitschriften eine andere, als bei Büchern oder im Kontext von Forschungsdaten. Und auch innerhalb desselben Dokumenttyps wird sich die Auswahl projektweise unterscheiden. Bei sich ändernden Anforderungen an die Suche müssen die Mechanismen zur Datenextraktion erweitert werden und der Betrieb würde mit Weiterentwicklungsaufgaben überfordern. Im Projekt wurde sich auf einen Basismetadatensatz (s. 4.4 Metadaten) verständigt, der beim Import geliefert werden muss.

Es wird jedoch davon ausgegangen, dass Archivobjekte bereits in **Recherche-, Katalog- oder Nachweissystemen** eingetragen sind. Diese Systeme bieten gute Suchfunktionen. Da der Entwicklungsaufwand für eine Suchschnittstelle groß ist, sollte das Archiv diesen nicht erneut betreiben, sondern auf die bestehenden Systeme aufbauen. Die genannten Systeme kennen den Anwendungskontext und die dort gebräuchlichen Metadaten, und sorgen sich darum, die Daten aktuell zu halten. Mit der über diese Systeme gefundenen PID oder der Referenz auf das Archivobjekt (PURL), kann ein Zugriff auf die Versionen im Archiv erfolgen. Die Suche sollte deshalb vornehmlich über die genannten Systeme erfolgen.

Das OLA-HD bietet eine einfache Metadatensuche. Zudem werden alle Volltexte beim Import indexiert, so dass auch eine Volltextsuche möglich ist.

3.3 Ergänzende Anforderungen

Ergänzende Anforderungen, wie z.B. die Adressierung von Archivobjekt Fragmenten, die Vermeidung von Redundanz, das Einsammeln (Harvesting) von Daten, die Bildung von Korpora, erweiterten Such- und Navigationsfunktionen, dem Export von Bestandteilen (einzelne Dateien), von Strukturelementen (Kapitel, Ansatz, Seiten, etc.), nach bestimmten Kriterien (von Autor, aus Jahr, etc.) erzeugt oder auch dem Export von Zitationsdaten, können direkt vom Archiv oder von externen Diensten realisiert werden

Insbesondere die aus Nutzerwünschen abgeleiteten ergänzende Anforderungen und damit verbundenen Funktionen erfordern eine genaue Kenntnis der Import- und Archivdaten, was einem generischen Archiv Ansatz entgegensteht, da dieses Wissen i.d.R. nur für einen konkreten Kontext vorliegt, oder umgekehrt nicht allgemeingültig festgelegt werden kann.

3.3.1 Identifizierung von Fragmenten und Duplikaterkennung

Die **persistente Adressierbarkeit und Zitierbarkeit von Versionen eines Textes, sowie von Fragmenten** unterhalb von bibliographischen Einheiten, ist von der Identifizierung dieser Einheiten abhängig und war eine der Projektanforderungen. Grundsätzlich greift die archivseitige Zuweisung von PIDs zu spät. Die Zuweisung von PID auf den unterschiedlichen Ebenen (Korpora, Werk, Kapitel, Absatz, Satz, Wort, Bild, etc.) muss bereits bei der Digitalisierung erfolgen. Sollte das Archiv die Zuweisung vornehmen, dann müsste die zugewiesene PID an den Datenlieferanten zurückgespielt werden, der die Daten entsprechend anpassen muss. Im Extremfall führt diese Anpassung beim Datenlieferanten erneut zu einem Export und einer neuen Version. Zudem entstehen Entscheidungsprobleme, z.B., wenn sich die Seitenreihenfolge, durch Korrektur von Fehlern bei der Digitalisierung, ändert oder sich der Kontext eines Fragments ändert, das Fragment selbst aber unverändert bleibt (ein unveränderter Satz in einem geänderten Absatz). Die Entscheidung ob im Zuge der Änderung alle Elemente eine neue PID zugewiesen bekommen müssen, kann das Archiv nicht treffen. Im Detail würde sich die PID Zuweisung im Archiv zu einer komplexen Aufgabe entwickeln. Die nachgelagerte Anpassung von Daten, d.h. außerhalb des Digitalisierungsprozess, passt nicht zu bestehenden Digitalisierungsworkflows. Eine Zuweisung von PIDs zu Fragmenten im Rahmen der Digitalisierung könnte bspw. erfolgen, indem im METS den jeweiligen XML-Div-Elementen der logischen oder physischen Strukturbeschreibungen, über das CONTENTIDS Attribut eine PID zugewiesen wird. Dieser Vorschlag wurden an die relevanten Arbeitsgruppen, z.B. bei “Kitodo. Key to digital objects” e. V. und der Deutschen Digitalen Bibliothek zurückgemeldet.

Wenn die Fragmente unterhalb von bibliographischen Einheiten, z.B. Kapitel, Absätze, Zeilen, Wörter oder Zeichen, nicht bereits bei der Digitalisierung eine PID zugewiesen bekommen haben, dann gibt es unterschiedliche Optionen um eine Identifizierung und Referenzierung dennoch zu ermöglichen. Bei der METS-basierten Beschreibung, wie sie in OCR-D zum Einsatz kommt (s. 4 Daten), ist die Identifizierung logischer Strukturelemente (z.B. Kapitel, Absatz, Inhaltsverzeichnis) bspw. mit deren XML-Element ID möglich. So könnte das Fragment durch Kombination von PID und Strukturelement ID (PID#LogID) identifiziert oder durch

Kombination von PURL und Strukturelement ID (PURL#LogID) referenziert werden. Fragmenten unterhalb der erwähnten Strukturelementen (z.B. Sätze, Zeilen, Wörter oder Zeichen) können bspw. dadurch identifiziert werden, dass die Einheiten als Sequenz durchnummeriert aufgelistet werden. Mit der Position in dieser Auflistung kann wiederum eine zusammengesetzte ID aufgebaut werden. Dabei muss der Kontext (Seite oder Gesamtdokument) klar festgelegt sein. Im Detail können aber viele Probleme bei einer derartigen Identifizierung von Sätzen, Zeilen, Wörtern oder Zeichen auftreten. Zeilenumbrüche, Seitenumbrüche oder Worttrennungen können zu komplexen Herausforderungen führen. Für die Identifizierung kann auch die ID der XML-Elemente im OCR oder die Koordinaten des zu identifizierenden Elements verwendet werden. Für physische Elemente, d.h. Dateien, können Prüfsummen berechnet und gespeichert und für die Identifizierung und Referenzierung verwenden werden. Bei logischen Elementen (Kapitel, Absatz etc.) ist die Berechnung von Prüfsummen erst mit qualitativ hochwertigen OLR-Ergebnissen (Optical Layout Recognition) möglich. Bei kleineren Einheiten, z.B. auf Wortebene, ist die Identifizierung durch Prüfsummen nicht mehr möglich, da z.B. alle Vorkommen eines Wortes im Text mit derselben Prüfsumme Identifiziert würden.

Da Fragmente nicht direkt mit einer ladbaren Ressource assoziiert sind, muss das Anwendungssystem, auf das der Resolver nach Auflösung der PURL weiterleitet, die Fragment ID bzw. Part ID (z.B. die erwähnte LogID) auflösen. Das Anwendungssystem öffnet bspw. eine spezielle Ansicht, oder liefert ein Dokument.

Das Software Heritage Archive[17] archiviert z.B. Software-Code der im GitHub[18] gespeichert ist. Über eine spezielle Ansicht kann eine Code-Zeile oder ein Code-Block markiert werden. Zu dem markierten Bereich kann ein Permalink (eine PURL) erzeugt werden. Mit dem erzeugten Permalink wird die aktuell geöffnete Seite, inklusive der Markierung, dauerhaft Identifiziert und Referenziert, d.h. sie kann darüber mit der gesetzten Markierung wieder geöffnet werden. Der Permalink beinhaltet dafür Kontextinformation, z.B. für die Markierung PURL#L19-L20 oder PURL;lines=19-20. Die Identifier werden im Software Heritage Archive durch Prüfsummen gebildet, z.B. die Prüfsumme einer Datei (vgl. Identifiers for Digital Objects: the Case of Software Source Code Preservation[19]). Die berechnete Prüfsumme wird zudem zur **Duplikaterkennung und -vermeidung** eingesetzt. Wenn der Speicherort durch die Prüfsumme bestimmt wird, dann werden Duplikate nur einmal gespeichert, denn jede identische Kopie derselben Datei hat dieselbe Prüfsumme. Damit wird eine deutliche Optimierung des Produktivspeicherbedarfs erzielt. Im OLA-HD ist die Markierung und entsprechende Referenzierung von Volltextzeilen und -bereichen aktuell nicht möglich, da die Benutzungsschnittstelle sehr einfach ist und nur die Anzeige von Bilder und Volltexte unterstützt. Eine Adressierung der betreffenden Datei ist über die Prüfsumme der Datei oder durch eine Kombination aus PID und Dateipfad möglich, so dass das OLA-HD die Funktion bei Erweiterung der Benutzungsschnittstelle anbieten kann.

[17] https://archive.softwareheritage.org

[18] System zur Versionsverwaltung und zum Source-Code-Management in Softwareprojekten

[19] https://hal.archives-ouvertes.fr/hal-01865790v4/document

3.3.2 Duplikaterkennung

Versionen unterscheiden sich i.d.R. durch kleine Details, d.h. geänderte, neu hinzukommende oder gelöschte Teile. Da der Großteil der Importdaten stabil bleibt, wird durch Duplikate unnötig Speicherplatz belegt. Durch **Duplikaterkennung** kann man dies deutlich verringern und zumindest die Plattennutzung des Produktivspeicher optimieren.

Um dies relativ einfach erreichen zu können, werden Dateien nicht unmittelbar unter einer Version und dem Dateinamen gespeichert, stattdessen wird der Name und damit der Ablageort durch die Prüfsumme der Datei bestimmt. Wenn sich eine Datei nicht ändert, ist die Prüfsumme dieselbe und die Datei wird nur einmal gespeichert. Es ist jedoch zu beachten, dass eine Datei ggf. zu verschiedenen Versionen gehört, was aus dem Dateinamen bzw. Ablageort nicht mehr ersichtlich ist.

Die prüfsummen-basierten IDs sind nicht auf lokal gespeicherte Objekte beschränkt. Wenn ein verteiltes System einen gemeinsam genutzten key-basierten Objekt Speicher (z.B. S3) einsetzt, und die Prüfsumme als Objekt ID (S3 Key) verwendet wird, so ist eine **redundanzfreie Speicherung** garantiert. Inhaltlich identische Dateien werden damit nur einmal gespeichert. Das gilt sowohl für Einzeldateien, als auch für ein gepacktes Archivobjekt.

Um die Zugehörigkeit zu Versionen sichtbar zu halten, wird z.B. durch eine Kombination aus klassischer, direkter Ablage im Dateisystem und der Prüfsummen-basierten Speicherung erreicht. Versionen bilden einen eigenen Pfad in der Hierarchie. Das Archivobjekt wird dabei gemäß der Objekthierarchie auf eine Verzeichnisstruktur auf der Festplatte abgebildet, wobei konkrete Dateien durch Verweise (Harte Verknüpfungen, bzw. Hard Links) ersetzt und die Inhalte in den nach Prüfsumme benannten Dateien gespeichert werden, auf welche die Verweise zeigen. Moderne Dateisysteme können den zuvor beschriebenen Mechanismus der **Deduplizierung** direkt unterstützen.

Der zuvor beschriebenen Ansatz zur Duplikaterkennung wird vom OLA-HD verwendeten Archiv-Manager (s. 2.6 Archiv-Manager CDSTAR) unterstützt, Archivobjekte werden damit auf der Platte nur einmal gespeichert und ggf. von verschiedenen Stellen aus referenziert.

3.3.3 Einsammeln der Daten von verschiedenen Orten

Mit dem **Harvesting von (Meta-)Daten** wird ein Mechanismus zum Bezug und Aktualisieren von Daten beschrieben. Das Open Archives Initiative Protocol for Metadata Harvesting (OAI PMH) [20] beschreibt eine XML- und REST-basierte Standardschnittstelle, mit der Datenlieferanten Datenänderungen veröffentlichen können. Datennutzer können Änderungen an den Daten für einen bestimmten Zeitraum von einem OAI PMH Server abfragen und ihren Datenbestand damit aktualisieren. Der Server muss mindestens eine Liste mit Dublin Core Datensätzen liefern, i.d.R. sind PMH Server aber auch in der Lage, Listen mit METS/MODS Datensätzen zu liefern. Da das Einsammeln von Metadaten automatisch erfolgt, ist es eine komfortable Lösung zur Aktualisierung des Datenbestands.

Der Download von Daten bringt jedoch eine kritische Abhängigkeit zu einem externen System mit sich, die das Archiv nicht kontrollieren kann. So kann der Service bspw. unerreichbar sein,

[20] https://www.openarchives.org/pmh/

oder das System limitiert die Anzahl der Downloads. Es entsteht bei großen Importvorhaben eine erhebliche Netzwerklast, da Einzeldateien geladen werden. Zudem kann die Integrität und Authentizität der geladenen Daten u.U. nicht geprüft werden, weil die Prüfsummen der Dateien nicht vorliegen, Datenfehler würde erst auffallen, wenn die Daten genutzt werden.

Neben dem systematischen sammeln von Metadaten für eine größere Menge von Archivobjekten, besteht auch die Möglichkeit, Metadaten für einzelne Archivobjekte aus Katalogen, Nachweissystemen oder Datenbanken auszulesen. Das kann z.B. für die VD18 Daten (Das Verzeichnis Deutscher Drucke des 18. Jahrhunderts), wie sie in OCR-D für Demonstrationszwecke zum Einsatz kommen, über die VD18 Datenbank[21] erfolgen. Solange der Anwendungskontext eingeschränkt ist, lassen sich Metadaten so gut ermitteln. Problematisch ist jedoch, dass die Adresse z.B. der Datenbank bekannt sein muss, und das jedes System potentiell eine andere Anfragesprache verwendet, mit anderer Abfragelogik. Zudem sind Kontexte denkbar, wo es noch keine entsprechenden Systeme gibt, z.B. für Forschungsdaten. Ein weiteres Problem ist erneut die Abhängigkeit zu externen Systemen, die nicht kontrolliert werden kann. Wenn z.B. die Datenbank vorübergehend nicht erreichbar ist oder den Betrieb einstellt hat, dann funktioniert der Metadatenbezug nicht. Das gleiche gilt für Änderungen an der Schnittstelle. In solchen Situationen entsteht ein erheblicher Wartungsaufwand, den ein Archiv nur schwer leisten kann. Daher wurde sich im Projekt entschieden, die Lieferung mit einem Basismetadatensatz zu fordern, so dass wichtige Metadaten mit dem Import vorliegen.

Für das OLA-HD wird das Einsammeln von Metadaten nicht genutzt, da die Datenproduzenten nicht über eine entsprechende Schnittstelle verfügen und wichtiger noch, da OAI PMH vornehmlich für den Austausch von Metadaten gedacht ist. Im OLA-HD müssen aber neben Metadaten auch die Nutzerdaten (Bilder, OCR, etc.) gespeichert werden. Wenn die OAI PMH Schnittstelle Metadaten im Format METS/MODS liefert (s. 4.5.1 METS) liegen die erforderlichen Informationen vor, um Nutzerdaten von der im METS referenzierten Stelle laden zu können. Erschwerend kommt hinzu, dass die Daten archiv-seitig gepackt werden müssen, um sie im Archivspeicher abzulegen. Mit dem im Projekt entwickelten Packaging-Werkzeug[22] ist das jedoch ohne weiteres möglich.

Im OLA-HD wird die aktive Datenübergabe des SIP vom Datenlieferanten an das Archiv gefordert, das Archiv ist passiv und nimmt die Daten an der Schnittstelle entgegen. Der Datenlieferant ist für den Import verantwortlich, er hat den Status des Datenimports zu überwachen und muss den Import bei einem Importfehler ggf. wiederholen.

3.3.4 Bildung von Korpora

Das Zusammenfügen mehrerer bibliographischer Einheiten zu Korpora kann auf unterschiedlichen Ebenen durchgeführt werden. So kann eine facettierte Suche eine Sicht ermöglichen, die eine Gruppierung simuliert. Das ist auch ohne besondere Funktionalitäten des Archivs möglich. Die Suche kann jedoch nur über bestehende Metadaten laufen. Damit können aber nicht benutzerdefinierter (virtueller) Kollektionen aufgebaut werden. Um letzteres zu ermöglichen, muss das System die Auszeichnung (Annotation) von Archivobjekten

[21] https://kxp.k10plus.de/DB=1.65/

[22] https://github.com/subugoe/OLA-HD-IMPL/tree/master/packaging-tool

ermöglichen, womit Archivobjekte quasi von außen um Metadaten erweitert werden. Dabei kann es sich um beliebige Metadaten handeln. Im vorliegenden Fall wird eine Beziehung zwischen dem Archivobjekt und dem Korpus aufgebaut, was mit Mitteln des Ressource Description Framework (RDF) [23] oder Annotationen [24] erfolgen kann. Das Beispielsystem Fedora Commons (s. 5.1.6 Fedora Commons) verwendet zur Datenverwaltung ein graph-basiertes Datenmodell. Hier ist es einfach möglich, eine Ressource per RDF mit zusätzlichen Metadaten bzw. Annotation auszustatten und darüber bspw. die Zugehörigkeit zu einem Korpus auszudrücken. Der Ansatz ist jedoch auch auf andere System anwendbar, indem ein Triplestore eingebunden und darüber Beziehungen beschrieben werden. Die Suche erfolgt in einer speziellen Abfragesprache: SPARQL[25].

Das OLA-HD unterstützt nicht die Bildung von virtuellen Kollektionen.

3.3.5 Such- und Navigationsfunktionen

Da das Archiv Daten über eine sehr lange Zeit speichern soll, ist davon auszugehen, dass das Archiv die Speichersysteme und Repositorien der Datenlieferanten überdauern wird. Ab einem bestimmten Zeitpunkt wird das Archiv das einziges System sein, in dem die Daten gespeichert und zugreifbar sind, womit das Archiv eine basale Suchschnittstelle anbieten muss, um Archivobjekte finden zu können.

Eine erweiterte Suche mit Filtermöglichkeiten und die Einrichtung von Navigationsmöglichkeiten, erfordert zum einen eine tiefe Kenntnis der Metadaten, die Übertragung der Metadaten auf entsprechende Indexstrukturen, sowie die Einrichtung von Navigationskomponenten in der Präsentationsschicht. Diese Kenntnis der Metadaten wird bei einem generischen Archiv kaum vorliegen und eine erweiterte Suche über einen Basismetadatensatz (s. 4.4 Metadaten), ist unnötig. Eine Navigation über Archivobjekte erfordert eine entsprechende Präsentationsschicht, über die ein Archiv nicht verfügt.

Das OLA-HD unterstützt eine basale Suche, eine erweiterte Suche und Navigation, wie zuvor beschrieben, wird aber nicht angeboten. Es ist jedoch zu einem späteren Zeitpunkt möglich die Funktionalität zu ergänzen und den Datenbestand dafür erneut zu indexieren.

3.3.6 Schnittstellen

3.3.6.1 Export von Bestandteilen

Der Bedarf nach einem Export von Bestandteilen von Archivobjekten erscheint im Kontext von OCR-D durchaus plausibel. So ist es denkbar, dass für die erneute Prozessierung die hochauflösenden Bilder, die Original METS Datei und ggf. Ground Truth Daten erforderlich sind.

Das OLA-HD ermöglicht dies über eine Auswahlkomponente der Benutzungsschnittstelle (s. 3.2.8.1 Benutzungsschnittstelle).

[23] https://www.w3.org/RDF/

[24] http://www.openannotation.org/spec/core/

[25] https://www.w3.org/TR/sparql11-query/

3.3.6.2 Export von Strukturelementen

Neben dem vollständigen Export eines Archivobjekts, sind spezifische Exportvarianten denkbar, die Nutzer für konkrete Analysen benötigen, z.B. der Export

- logischer (Band, Kapitel, Unterkapitel, etc.) und physischer Strukturelemente (Seitenbereiche),
- aller Bilder einer Einheit (logisch oder physisch) als PDF Dokument,
- aller Volltexte einer Einheit, z.B. als TEI oder HTML Dokument,
- als IIIF Manifest für Präsentationszwecke, oder
- als Zitationsdaten (z.B. Bibtex, Endnote, RIS).

Beim ersten Punkt stellt sich schon die Frage, was vom Strukturelement, wie exportiert werden soll. Sollen die Bilder exportiert werden, oder die Volltexte? Ein Export als PDF würde eine Konvertierung erfordern, was wiederum die Kenntnis der Struktur erforderlich macht.

Im Kontext von OCR-D ist der Bedarf nach ergänzenden Exportfunktionen nicht gegeben, so dass das OLA-HD diese Funktionalität nicht unterstützt.

3.3.6.3 Export nach bestimmten Kriterien

Der Export nach bestimmten Kriterien, hat eine Filterung oder Beschränkung des Exportes zum Ziel. Kriterien dafür könnten z.B. sein, dass der Export

- nur die Seiten 1-20 für alle Dateigruppen (Bilder, OCR, etc.) umfassen soll, oder
- alle Werke umfasst, die
    - von einem bestimmten Autor stammen, oder
    - ein bestimmtes Veröffentlichungsdatum haben, oder
    - in einer bestimmten Sprache verfasst sind.

Beim Export nach bestimmten Kriterien handelt es sich um einen Export in Abhängigkeit einer erweiterten Suche.

Auch diese Funktionen werden im Kontext von OCR-D nicht benötigt. Das OLA-HD unterstützt diese Funktionen nicht.

4 Daten

4.1 Vorbemerkung

Daten müssen in einem digitalen Archiv über lange Zeiträume gespeichert werden und auch in Zukunft nutzbar sein. Ein heute übliches Format kann schon in wenigen Jahren obsolet sein. Es kann nicht garantiert werden, dass ein bestimmtes Format zukunftsfähig oder zumindest migrationsfähig ist. Mit PDF/A liegt zwar ein zukunftsfähiges Format vor, jedoch ist dies für die gängigen Präsentationswerkzeuge, die ausschließlich Bilder anzeigen können, ungeeignet

und auch in den DFG-Praxisregeln "Digitalisierung"[26] wird von der Verwendung von PDF/A abgeraten, da "deren Nachnutzung gerade durch die digitale arbeitenden Geistes- und Kulturwissenschaften mangels struktureller Auszeichnung beschränkt ist".

Um Daten zukunftsfähig zu halten, sind **Formattransformationen** erforderlich, die Daten in ein interpretierbares Format überführen. Neben der Kenntnis von Ursprungs- und Zielformaten sind Formattransformationen festzulegen und zu implementieren. Diese Aufgabe ist komplex und aufwändig, da sie verschiedenste Formate berücksichtigen muss und sie ist nie abgeschlossen. Das macht die Implementierung, Wartung und Pflege eines digitalen Archivs sehr aufwändig. Mit jedem neuen Format steigt die Komplexität der Aufgabe. Formattransformationen erfordern die Festlegung von bzw. Einschränkung auf konkrete Formate und sie stehen dem generischen Anspruch entgegensteht.

Die LZA adressiert Formattransformationen (als Teil des Preservation Planning, Technology Watch), um frühzeitig aktiv werden zu können und die Daten stets auf dem aktuellen Stand der Technik zu halten. Eine bedarfsgeleitete Implementierung und Transformation kann demgegenüber zu spät greifen, d.h. zu einem Zeitpunkt, bei dem die Daten nicht mehr interpretierbar sind.

Der Aspekt der Formattransformation wird vom OLA-HD, wegen der begrenzten Projektlaufzeit und Ressourcen nicht behandelt.

4.2 Importdaten

Daten durchlaufen von der Produktion bis zur Archivierung verschiedene Phasen: Produktion, Nachverarbeitung, Auswertung, Präsentation oder Archivierung. In den Phasen stehen unterschiedliche Datenformate im Mittelpunkt, die aber nicht zwangsläufig archiviert werden müssen. **Abgeleitete Daten und Dokumente** (Derivate), wie bspw. Bilder in geringen Auflösungen, IIIF-Manifeste oder PDFs, können aus Originaldaten erzeugt werden. Ein Import dieser Daten ist nicht zwingend erforderlich. Dennoch kann ein Import aus Komfortgründen angemessen sein, bspw. wenn das Archiv eine Bildvorschau anbieten will. Die Erzeugung der Bilder zur Laufzeit ist mit einer zu großen Antwortzeit für den Benutzer verbunden.

Im OLA-HD wird neben den hochauflösenden Bildern auch eine einfache Auflösung gespeichert, die eine Vorschau der Bilder ermöglicht.

4.3 Austauschformat

4.3.1 OCRD-ZIP

In OCR-D wurde OCRD-ZIP[27] als Austauschformat für Importpakete festgelegt. OCRD-ZIP baut auf BagIt auf und nutzt ZIP als Paketierungsformat. Die BagIt Spezifikation[28] bildet einen konzeptionellen Rahmen für Nutzerdaten und beschreibende Metadaten des Importpaketes,

[26] https://www.dfg.de/formulare/12_151/12_151_de.pdf

[27] https://ocr-d.de/de/spec/ocrd_zip

[28] https://tools.ietf.org/html/draft-kunze-bagit-16

macht aber keine Vorgaben oder Einschränkungen zu den Nutzerdaten. Im Importpaket werden mit den Nutzerdaten auch Prüfsummen, zu jeder Einzeldatei und zum Importpaket als Ganzes, gespeichert, wodurch die Integrität bzw. Korrektheit des Importpaketes sichergestellt werden kann. OCRD-ZIP fordert, dass die Datei zur Metadatenbeschreibung über das Importpaket (Bag Metadata: bag-info.txt) verpflichtend ist, in der BagIt Spezifikation ist sie optional. Somit müssen wichtige Metadaten nicht aus verschiedenen Dokumenten gelesen werden, sondern stehen immer an festgelegter Stelle zur Verfügung.

Es wird zudem gefordert, dass alle Daten im Importpaket enthalten sind. Der Vorteil der Abgeschlossenheit des Importpaketes, der immer auch eine vollständige Version darstellt, hat den Nachteil, dass das Datenvolumen durch Lieferung vollständiger Archivobjekte immer groß ist, auch wenn eine neue Version eingeliefert wird, bei der sich nur ein kleiner Teil geändert hat. Ein Import in Teilen ist nicht vorgesehen.

4.3.2 OCRD-METS

In OCR-D wurde zudem festgelegt, dass die **Beschreibung der Prozessierungsobjekte** (Importdaten) im METS Format erfolgt, wobei mit der OCRD-METS [29] Konvention Konkretisierungen vorgenommen werden. Mit METS können die Bestandteile eines digitalen Objekts (beschreibende, administrative und strukturelle Metadaten, sowie Beziehungen) beschrieben werden. In Kombination mit OCRD-ZIP wird ein Importpaket spezifiziert, dass eine vollständige Beschreibung eines digitalen Archivobjekts darstellt und alle zum Objekt zugehörigen Dokumente beinhaltet.

Auf METS und OCRD-METS wird nochmals im Abschnitt zu den Metadatenstandards eingegangen (vgl. 4.7 Metadatenstandards).

4.4 Metadaten

Im Projektverlauf wurde sich mit den Projektpartnern darauf verständigt, einen minimalen **Basismetadatensatz** zu fordern, der die von OCRD-ZIP geforderten Metadaten ergänzt und der in die Metadatenbeschreibung (bag-info.txt) mit aufzunehmen ist. Importe sollen danach mindestens eingeliefert werden mit den von OCRD-ZIP geforderten Metadaten, ergänzt um einen verpflichtenden Ausschnitt der Dublin-Core DCMI Metadata Terms[30]: DC.identifier, DC.title, DC.rights, DC.license. Der Satz wird ergänzt um einen optionalen Ausschnitt der DCMI Metadata Terms: DC.creator, DC.publisher, DC.location, DC.issued.

Die Metadatenauswahl muss als **kleinster gemeinsamen Nenner** verstanden werden. Die Anforderungen an Daten und Metadaten sind sehr projektspezifisch und liegen teilweise weit auseinander, oder es bestehen inkonsistente Schnittmengen, d.h. scheinbar identische Metadatenfelder sind mit unterschiedlicher Bedeutung belegt. So sieht ein Projekt die Daten bereits mit einer ID, einem Titel und einer Lizenzangabe ausreichend beschrieben. Für ein anderes Projekt mag demgegenüber der begrenzte Umfang als vollständig ungeeignet erscheinen, und allein das Metadatum "Titel" kann Diskussionen nach sich ziehen, weil es zu unspezifisch ist. Im Projekt wurde sich dennoch auf den überschaubaren Satz beschränkt, da

[29] https://ocr-d.de/de/spec/mets

[30] https://www.dublincore.org/specifications/dublin-core/dcmi-terms/

davon ausgegangen wird, dass die Daten bereits in projektspezifischen Repositorien oder Nachweissystem vorgehalten werden. Die Suche oder Recherche erfolgt über diese Systeme, denn ein Archiv ist nicht als Recherchesystem ausgelegt. Mit einer über das Nachweissystem ermittelten PID oder der dort gespeicherten Referenz (PURL) kann der Zugriff auf das Archivobjekt durchgeführt werden. Mit den geforderten Metadaten, ggf. ergänzt um die indexierten Volltexte der Archivobjekt Inhalte ist dennoch eine direkte **Suche im Archiv** möglich, was insbesondere dann zum Tragen kommt, wenn das ursprünglich verwendete Repositorium oder Nachweissystem abgeschaltete wurde.

4.5 Metadatenstandards

4.5.1 METS

Der Metadata Encoding & Transmission Standard (METS)[31] ist ein im Bibliothekskontext etabliertes XML-basiertes Beschreibungsformat für digitaler Objekte (Bücher, Zeitschriften, Zeitschriftenbände, etc.). Mit METS können die Bestandteile eines digitalen Objekts (z.B. Bilder, Ground Truth oder OCR), die logischen und physischen Strukturen, sowie ihre Beziehungen untereinander abgebildet, und mit bibliographischen und administrativen Metadaten beschrieben werden. METS legt dafür unterschiedliche Sektionen fest:

- dmdSec: **Beschreibende Metadaten** für das Gesamtobjekt oder auch einzelne Bestandteile (z.B. bibliographische Metadaten)
- amdSec: **Administrative Metadaten** für das Gesamtobjekt oder einzelne Bestandteile (z.B. Rechteinhaber oder Lizenz)
- fileSec: In **Dateigruppen** aufgeteilte Auflistung konkreter Dokumente, mit Verweis auf einzelne **Dateien** (Bilder unterschiedlicher Auflösungen, GT, OCR, PDF, etc.)
- structMap: Liegt in 2 Ausprägungen vor
    - logische Struktur: Bildet das **Inhaltsverzeichnis** bzw. die logische Struktur eines digitalen Objekts ab (es kann mehrere geben)
    - physische Struktur: **Seitensequenz** der physischen Dokumente
- linkSec: Aufbau von **Beziehungen** zwischen Elementen der logischen Struktur mit den Elementen der physischen Struktur, wodurch physischen Elemente einem logischen Element zugeordnet werden (z.B. log4 -> phys11...phys20, d.h. das logische Element log4 beginnt mit Seitenelement 11, …, Seitenelement 20 ist das letzte Element)[32]

Im METS gibt es i.d.R. keine Hinweise auf Versionen, was bedeutet, dass die Versionierung ausschließlich archiv-seitig stattfindet. Technische Metadaten oder Metadaten über die Entstehungshistorie (s. 4.5.4 ProvOne) eines digitalen Objekts können direkt im METS aufgeführt oder durch Referenzierung beschreibender Dokumente integriert werden. Referenzierte Dateien können sowohl lokal, d.h. relativ zur METS Datei abgelegt sein, als auch

[31] http://www.loc.gov/standards/mets/

[32] Ein Element der physischen Strukturbeschreibung muss nicht zwangsläufig eine Seite sein, es ist damit auch möglich Audiosequenzen innerhalb einer Audiodatei zu beschreiben.

als Referenz per URL eingebunden werden. Die lokale Ablage wird mit OCRD-ZIP festgeschrieben, so dass die Daten Teil des Importpaketes sind.

Bei der Auswahl und Integration der Metadaten sollte Benutzbarkeit (Verarbeitbarkeit und Performance) im Vordergrund stehen, die Komplexität sollte gering gehalten werden. So werden i.d.R. bibliographische Metadaten durch MODS (vgl. MODS) beschrieben und direkt in METS eingebettet. Das ist auch für die anderen, im folgenden einzuführenden Metadatenformate (z.B. PAGE XML, ProvOne, PREMIS, etc.) möglich. Mit der Einbettung weiterer Metadaten würden jedoch die ohnehin komplexen METS Dokumente deutlich umfangreicher und die interne Beziehungsstruktur noch komplexer. Es wurde sich deshalb im Projekt darauf verständigt, neben MODS keine weiteren Strukturen direkt im METS einzubetten. Die anderen Metadaten werden als eigenständige Dokumente gespeichert und aus METS heraus referenziert.

Die OCRD-METS Konvention nimmt Einschränkungen bzgl. Importdaten und Metadaten vor. So wird bspw., festgelegt, dass:

- die Bildauflösung mindestens 150 ppi (Pixel per Inch) betragen muss, und dass die Auflösung explizit in den Bildmetadaten ausgewiesen sein muss,
- Bilder als Single-Page Images vorliegen müssen, da die OCR-D Prozessoren keine Multi-Page Images verarbeiten können und Fehler verursachen,
- in den MODS Metadaten ein eindeutigen Identifier ausgewiesen werden muss, der vom Typ "purl", "urn", "handle" oder "url" ist,
- alle Dateien einer Dateigruppe denselben MimeType haben müssen,
- die USE-Attribute der Dateigruppen nach einem festgelegten Schema aufgebaut sein müssen, so dass aus dem Bezeichner hervorgeht, ob es sich bei den Dokumenten einer Gruppe um Ground Truth Daten, Bilder, Segmentierungsdaten (Region, Line, Word), OCR Dokumente oder Korrektur Daten handelt (ein Beispiel für eine Dateigruppe mit Bildern ist USE="OCR-D-**IMG**"),
- der Aufbau der Datei ID muss dem Aufbau des vorgenannten USE-Attribut folgen und mit einer null-aufgefüllten 4-stelligen Ziffer Enden (z.B. ID="OCR-D-**IMG_0001**"), oder dass
- Dokumente auf derselben Verzeichnisebene wie die der METS Datei oder in Unterverzeichnissen abgelegt sein müssen, und dass die Referenzierung, bspw. aus den METS Datei Elementen, stets relativ zur METS Datei erfolgen muss.

Diese Einschränkungen müssen von den Datenproduzenten beachtet werden.

4.5.2 MODS

Das Metadata Object Description Schema [33] ist ein XML Schema zur Beschreibung bibliographischer Metadaten. Die Beschreibung wird meist zusammen mit METS verwendet

[33] https://www.loc.gov/standards/mods/

und dort direkt in die Sektion für die beschreibenden Metadaten (dmdSec) unter einem eigenen XML-Namespace integriert.

OCRD-METS fordert lediglich einen eindeutigen Identifier, so dass der Umfang an beschreibenden Metadaten relativ frei bestimmt werden kann.

4.5.3 OCR Standards und Formate

4.5.3.1 Alto

Der Analyzed Layout and Text Object (ALTO)[34] ist ein XML-basierter Standard, der eingesetzt wird, um in der OCR Prozessierung Text und Layout Informationen, sowie Metadaten über die Prozessierung zu beschreiben (vgl. GitHub Dokumentation[35]). Da die Beschreibung werkweise erfolgt, referenziert die entsprechende METS Dateigruppe nur eine Datei. Der Einsatz von ALTO wurde in den DFG-Praxisregeln "Digitalisierung" [36] empfohlen. ALTO Dokumente bestehen aus drei Sektionen:

- Description: **Beschreibende Metadaten** über die Datei und Prozessierunginformationen werden in Kindelementen beschrieben
- Styles: **Formatierungsinformationen** werden weiter Untergliedert
    - TextStyle: Font- und Schrifttypen
    - ParagraphStyle: Blockeigenschaften, z.B. linksbündig
- Layout: Enthält den **Seiteninhalt**

ALTO ist ein einfach zu implementierendes und interpretierendes Format, hat jedoch nicht die Detailtiefe wie PAGE XML.

4.5.3.2 hOCR

Das hOCR Format[37] ist ein HTML- bzw. XHTML-basiertes Beschreibungsformat (vgl. GitHub Dokumentation [38]). Mit hOCR können neben Text und Layoutinformationen auch Informationen über die Erkennungsrate dokumentieren werden. Metadaten werden in Dublin Core beschrieben und in Metadaten Sektionen des HTML eingebettet. Nachteilig bei dieser Ablage der OCR Ergebnisse ist, dass Informationen erst durch Attribute konkretisiert werden (z.B. class='ocrx_word' innerhalb eines HTML-span Block). Damit wird das extrahieren von Informationen erschwert. Der Einsatz von HTML hat den Vorteil, dass Dokumente direkt angezeigt werden können.

[34] https://www.loc.gov/standards/alto/

[35] https://github.com/altoxml/documentation/wiki

[36] https://www.dfg.de/formulare/12_151/12_151_de.pdf

[37] http://kba.cloud/hocr-spec/1.2/

[38] https://github.com/kba/hocr-spec

4.5.3.3 PAGE XML

PAGE XML[39] umfasst verschiedene XML-basierte Formate zum Erfassen von Text, Layout und Bildinformationen. Für OCR-D sind die Format zur Beschreibung von

- Text Informationen[40] (Region, Zeile, Wort, etc.),
- Layout Informationen[41] oder
- Bild Informationen[42] (Metadaten über erkannte Bilder).

von Interesse, wobei im OCR-D Projekt eine ergänzende Konvention (Conventions for PAGE[43]) erstellt wurde, welche die Verwendung in OCR-D konkretisiert.

PAGE XML ermöglicht eine sehr detaillierte OCR Dokumentation, wobei die Möglichkeit zur Abbildung von Bildinformationen hervorzuheben ist, was andere OCR Standards so nicht unterstützen. Die Detailtiefe geht mit einer Komplexität der Beschreibung einher, und es ist abzuwägen, ob ein einfacheres, leichter zu interpretierendes Format wie ALTO vorzuziehen ist.

Mit PAGE XML ist es möglich, Informationen über den OCR Prozess zu dokumentieren, z.B. durch benutzerdefinierte Metadaten oder Daten zur Software. Die Möglichkeiten sind jedoch sehr begrenzt und nicht so strukturiert, wie bei ProvOne (s.u.).

Die prozessierenden OCR-D Module erzeugen OCR Daten im PAGE XML Format. Die OCR Daten werden seitenbasiert und über eine eigene Dateigruppe im METS referenziert.

4.5.3.4 TEI

Die Text Encoding Initiative (TEI)[44] beschreibt XML-basierte Formate zur Textcodierung. Neben der Abbildung von Text, können Textstellen nahezu beliebig mit Zusatzinformationen ausgezeichnet werden. TEI ist grundsätzlich als Auszeichnungsformat für OCR Ergebnisse einsetzbar und in der Praxis werden die OCR Ergebnisse für die weitere Nutzung in Informations- und Präsentationssystemen nach TEI konvertiert. Mit der Konvertierung geht jedoch ein Informationsverlust einher, der nur mit großem Aufwand verhindert werden kann. In OCR-D steht der Vergleich von OCR Ergebnissen im Vordergrund. Wegen des Informationsverlustes bei der Konvertierung ist es zielführender, den Vergleich auf den Originaldaten (Bilder, PAGE XML) durchzuführen. TEI bietet sich daher im Kontext von OCR-D nicht als Masterformat an.

[39] https://github.com/PRImA-Research-Lab/PAGE-XML

[40] https://www.primaresearch.org/schema/PAGE/gts/pagecontent/2019-07-15/pagecontent.xsd

[41] https://www.primaresearch.org/schema/PAGE/eval/layout/2013-07-15/layouteval.xsd

[42] https://www.primaresearch.org/schema/PAGE/gts/dewarping/2014-08-26/dewarping.xsd

[43] https://ocr-d.de/de/spec/page

[44] https://tei-c.org

4.5.4 ProvOne

Das ProvONE Data Model for Scientific Workflow Provenance, eine Erweiterung des PROV Standards des W3C [45], ist dafür ausgelegt, Informationen über die Prozessierung zu dokumentieren. Die Erweiterung gegenüber PROV betrifft die Möglichkeit der Abbildung wissenschaftlicher Workflows. Das ist wichtig, da nie ein Einzelschritt zum Ergebnis führt. Es stehen nicht die Ergebnisdokumente im Vordergrund, sondern die Beschreibung des Weges dahin. Bei jedem OCR Prozessierungsdurchgang werden neue Metadaten erzeugt, welche Hinweise darauf geben, wie ein Zustand erreicht wurde. Mit ProvOne wird die Black-Box-Sicht auf die Prozessierung (Eingabe -> Prozessor -> Ergebnis) aufgebrochen. Prozessdaten können für das Gesamtobjekt oder für jede Einzelseite erzeugt werden. Erst mit der Verfügbarkeit der Provenienzinformationen können die unterschiedlichen Prozessierskonfigurationen detailliert beurteilt und verglichen werden. Es kann damit genau gesagt werden, wo diese oder jene Konfiguration besser ist und es wird einen Einstiegspunkt für Korrekturen geboten. Die Daten müssen als Teil des Importpaketes vom Datenproduzenten bereitgestellt werden, sie sind nicht nachträglich ermittelbar.

Zentrale Elemente des ProvOne Datenmodells sind die Objekte (Entities) zu denen Provenienzinformationen dokumentiert werden sollen, Personen und Organisationen, Dienste oder Softwarekomponenten (Agents), die Objekte geliefert oder manipuliert haben, Ereignisse oder Prozesse (Events) die zu einem neuen Objektzustand geführt haben. In ProvOne können zudem Prozessketten (Workflows) und Versionen direkt abgebildet werden. Elementen des Datenmodells werden IDs zugewiesen und sie werden über diese Verknüpft.

Die Abbildung der Provenienzinformationen ist allerdings nicht einfach und die eingesetzten Prozessierungswerkzeuge unterstützen ProvOne i.d.R. nicht. Da die Erzeugung der Beschreibungen bestenfalls halbautomatisch erfolgen kann, ist sie für die Projektmitarbeiter zudem aufwändig. Es wurde sich daher im Projekt gegen die Abbildung von Provenienzinformationen entschieden. In OCR-D muss jedoch berücksichtigt werden, dass Provenienzinformationen teilweise in den PAGE XML Dokumenten enthalten sind und das Zwischenergebnisse als Dokumente vorliegen (Ergebnisse der Bilderzeugung, Segmentierung, OCR oder Korrektur). PAGE XML erreicht zwar nicht die Detailtiefe von ProvOne, aber für die Fragestellungen und Vergleiche in OCR-D sind sie ausreichend.

Es bleibt jedoch zu empfehlen Provenienzinformationen explizit zu erheben, womit die Entstehung der Archivobjekte nachvollziehbar und ggf. sogar reproduzierbar wird.

4.5.5 PREMIS

Der PREMIS Data Dictionary for Preservation Metadata Standard (PREMIS)[46] ist ein Standard zur Beschreibung von Provenienzinformationen, d.h. Metadaten über die Entstehungs- bzw. Veränderungsgeschichte eines digitalen Objekts im Kontext der Langzeitarchivierung. Es werden damit Archivierungsmaßnahmen dokumentiert, z.B. Datenmigrationen oder Formattransformationen, die den Zustand eines Archivobjektes verändert haben.

[45] https://www.w3.org/TR/2013/NOTE-prov-overview-20130430/

[46] https://www.loc.gov/standards/premis/

Das PREMIS Datenmodell ist dem von PROV sehr ähnlich. Es betrachtet neben den bereits von ProvOne verwendeten Elementen Entities, Agents und Events auch die Rechte bzgl. dieser digitalen Objekte (Rights). Das ist wichtig, damit diese Informationen auch über Speichermigrationen und Formattransformationen hinweg erhalten bleiben.

Auch die Abbildung von Provenienzinformationen bzgl. der LZA ist nicht einfach und wird von den Prozessierungswerkzeugen nicht unterstützt, so dass es im Projekt nicht berücksichtigt wurde (s. Begründung in 4.5.4 PROV).

4.5.6 textMD

Beim Technical Metadata for Text (textMD)[47] Schema handelt es sich um ein XML-basiertes Beschreibungsformat um textuelle digitale Objekte zu beschreiben. So können Kodierungs- und Formatierungsinformationen, Sprache, Seitenstruktur, Prozessierungsinformationen, etc. dokumentiert werden. textMD wurde im Projekt als Option zur Abbildung der Prozessierungsinformationen und -historie diskutiert. Das Schema sieht jedoch nur ein XML-Element vor, dass die Prozessierung als Notiz zu beschreiben erlaubt und es gibt keine Vorgaben, welche Informationen in welcher Form abzubilden sind. Damit bleiben die Informationen unstrukturiert.

Für die Beschreibung textuellen Ressourcen ist PAGE XML oder ALTO besser geeignet. Zur Abbildung der Prozessierungsinformationen bieten ProvOne deutlich bessere Möglichkeiten.

4.6 Exportdaten

Die Erwartung der Benutzer ist, dass mindestens der Export der Importdaten möglich ist. Im OLA-HD erfolgt der Export im OCRD-METS Format aus Gründen der Antwortzeit standardmäßig aus dem Produktivspeicher. Im Produktivspeicher sind jedoch u.U. nicht alle Daten eines Archivobjekts enthalten. So werden dort bspw. hochauflösende Bilder nicht mitgespeichert, um die Speicherbelegung zu optimieren. Der unveränderte Export des vollständigen Archivobjekts ist aus dem Archivspeicher möglich, was aber wegen des Zugriffs auf den Bandspeicher mit einer größeren Antwortzeit verbunden ist. Der Export wird über die PURL des Archivobjekts durchgeführt.

Wie im Abschnitt 3.2.8.1 Benutzungsschnittstelle erwähnt, ist es zudem möglich, Bestandteilen eines Archivobjekts über eine Auswahlkomponente der Benutzungsschnittstelle anzuzeigen, für den Download zu markieren und als ZIP Archiv zu exportieren.

5 Evaluierung bestehender Systeme

Neben der Erfüllung der Anforderungen wie sie in Abschnitt 3 Anforderungen aufgezeigt wurden, ist Antwortzeit bei der Suche und die Upload- und Downloadzeit beim Import und Export ein Kernkriterium für die Auswahl eines Systems. Der Datendurchsatz und der Antwortzeit, sowie die Stabilität der Werte unter Last, kann durch messen konkret beurteilt

[47] https://www.loc.gov/standards/textMD/

werden. Variablen Größen, wie bspw. die Auslastung des Systems, die Netzwerkauslastung und -verfügbarkeit oder durch die Größe des Importpaketes, haben jedoch einen entscheidenden Einfluss auf Transfer- und Antwortzeiten. Insofern erscheint es angemessener, eine relative Performancebetrachtung vorzunehmen, um Aspekte hervorzuheben oder einen groben Vergleich vorzunehmen.

5.1 Auswahl bestehende Systeme

5.1.1 archivematica

Kurzbeschreibung: archivematica ist eine webbasierte Anwendung zum Archivieren von Dokumenten und Daten. Es verfügt über einen großen Funktionsumfang um den Zugriff langfristig zu sichern, es ist konform zum OAIS Referenzmodell und unterstützt direkt LZA Maßnahmen.
Metadaten: Dublin Core und PREMIS per METS. Dublin Core per Metadaten Datei.
Identifikation: Benutzerdefinierte IDs und vom System bezogene Handle PIDs.
Versionierung: Nein.
Zugriff: Auf Datenobjekte, Dateien und Metadaten.
Suche: Via ElasticSearch.
Import (gepackt): Ja (BagIt ZIP). Staging Bereich (Backlog).
Import (dateibasiert): Nein.
Export (gepackt): Ja, als BagIT ZIP. Es können verschiedene DIPs beschreiben werden, die eine eigene PURL haben.
Export (dateibasiert): Nein.
Möglichkeit zur Gruppierung/Kollektionsbildung: Ja, via Archival Information Collection (AIC). Die Archivobjekte bleiben voneinander unabhängig und werden über AIC referenziert und gruppiert.
API: Nur zum Approval.
Performance Import: Import selbst gut. Nachverarbeitung bzw. Konfiguration der Objekte im Backlog ist erforderlich und aufwändig.
Performance Export: Gut.
Performance Suche: Gut
Clusterisierung: Ist möglich.
Limit Anzahl der Objekte: Kein Limit.
Limit Speicher: Kein Limit. archivematica kann unterschiedliche Speichersysteme integrieren.
LZA Aspekte: Preservation Planning, z.B. Formatmigrationen. Metadaten der Langzeitarchivierung (z.B. Dokumentation von Formatmigrationen) werden vom System gepflegt und im PREMIS Format dokumentiert.
Authentifizierung: LDAP[48], Shibboleth[49].
Autorisierung: Nein.
Link: https://www.archivematica.org/
Besondere Eigenschaften:

- METS basiert, paketierter Import (BagIt ZIP)
- Einbindung Handle.net, Zuweisung von PID zu Archivobjekt, Erzeugung von PURL

[48] Lightweight Directory Access Protocol (LDAP), Netzwerkprotokoll zur Abfrage von Informationen aus Verzeichnisdiensten (https://tools.ietf.org/html/rfc4511)
[49] Konkretes SSO Verfahren zur verteilten Authentifizierung und Autorisierung.

- OAIS konform, Spezifikation von AIP und DIPs auf Basis des SIP über die Benutzungsschnittstelle
- Konzept der Zwischenablage (Backlog), in dem Importpakete nach Übernahme abgelegt werden, bis ihr Import beschrieben wurde, d.h. Abbildung auf API, ggf. Transformationen, Beschreibung von DIPs
- Vielfältige Funktionen über Dienste, z.B. Viruschecks, Formatcharakterisierung und -transformationen
- Verschiedene Importoptionen: via ZIP, Disk Image, aus DSpace oder DataVerse
- Re-Ingest des bereits importierten Importpaketes, so dass weitere Dienste auf dem Import durchgeführt werden können (Objekt wird neu prozessiert, nicht aber neu importiert)
- Approval Konzept
- Preservation Planning, automatische Erzeugung von Provenienzdaten und Abbildung auf PREMIS Datenmodell

Hervorzuhebende Nachteile:

- Keine API
- Keine Versionierung
- Konfiguration muss für jeden Import über die Benutzungsschnittstelle erfolgen, das ist aufwändig und nicht für große Importvorhaben geeignet

5.1.1 CDSTAR

Kurzbeschreibung: CDSTAR ist ein Datenmanagement System, das die Datenspeicherung und -suche von digitalen Datenobjekten (Container für alle zugehörigen Dateien und Metadaten) in wissenschaftlichen Projekten unterstützt und die Speicherbelegung optimiert.
Metadaten: Dublin-Core basierte und benutzerdefinierte Metadaten. Nur Dublin-Core Metadaten werden zur Unterstützung der Suche indexiert. Metadaten können sich auf ein komplettes Datenobjekt oder auf einzelne Dateien beziehen.
Identifikation: Systemgenerierte interne ID. Bezug und Zuweisung einer PIDs muss auf Applikationsebene vorgenommen und auf die Metadaten übertragen werden.
Versionierung: Jeder Import erzeugt ein neues Datenobjekt, das separat gespeichert wird. Versionierung muss auf Applikationsebene vorgenommen werden, indem in den Metadaten eine Referenz auf das Vorgängerdatenobjekt eingetragen wird.
Zugriff: Datenobjekts, Dateien und Metadaten.
Suche: Die Suche wird durch einen Suchindex (ElasticSearch) unterstützt. Suche und Navigation über Beziehungsgeflechte wird nicht unterstützt und muss auf Applikationsebene durchgeführt werden, z.B. mit Triplestore.
Import (gepackt): Nicht unterstützt. Entgegennahme und entpacken von ZIPs muss auf Applikationsebene durchgeführt werden.
Import (dateibasiert): Ein Datenobjekt wird nach und nach aufgebaut. Nachträgliche Ergänzungen (Dateien und Metadaten) sind möglich.
Export (gepackt): Nicht unterstützt. Muss auf Applikationsebene durchgeführt werden.
Export (dateibasiert): Direkter Zugriff auf Dateien, per Dateiprüfsumme, Bitstream ID, Dateipfad.
Möglichkeit zur Gruppierung/Kollektionsbildung: Nicht unterstützt.
API: Ja. Gut dokumentierte REST API.

Performance Import: Gut.
Performance Export: Gut.
Performance Suche: Gut.
Clusterisierung: Ist möglich.
Limit Anzahl der Objekte: Kein Limit.
Limit Speicher: Kein Limit. CDSTAR kann unterschiedliche Speichersysteme integrieren (S3, iRODS, hadoop, network storage).
LZA Aspekte: Backup und Bitstream Preservation werden von der genutzten Infrastruktur behandelt. Metadaten der Langzeitarchivierung (z.B. Dokumentation von Datenmigration) müssen auf Applikationsebene gepflegt werden.
Authentifizierung: LDAP, SSO[50] via Shibboleth
Autorisierung: Rollenbasierte Zugriffskontrolle (Benutzer, Gruppen, Rollen)
Link: https://cdstar.gwdg.de/
Besondere Eigenschaften:

- Optimierung der Speichernutzung durch prüfsummenbasierte Speicherung
- Metadaten auf allen Ebenen
- Gute API
- Gute Dokumentation

5.1.2 CLARIN Virtual Language Observatory (VLO)

Kurzbeschreibung: Recherche System, das eine facettierte Suche und die Navigation unterstützt. Das VLO speichert nur Metadaten und referenziert auf extern gespeichert Daten.
Metadaten: Metadaten müssen im Dublin-Core Format bereitgestellt werden.
Identifikation: Per PID oder PURL.
Versionierung: Nicht unterstützt.
Zugriff: Erfolgt per Referenz (PURL) auf die extern gespeicherten Daten, das kann die Startseite in einem Portal sein oder eine konkrete Ressource (z.B. ein Bild oder ein PDF).
Suche: Über die Benutzungsschnittstelle wird eine gut strukturierte und umfangreiche Suchfunktion (Facettierung, Navigation) angeboten, mit logischen Operatoren können Suchterme verknüpft werden.
Import (gepackt): Nicht unterstützt.
Import (dateibasiert): Der Import der Metadaten erfolgt ausschließlich per Harvesting durch OAI-PMH, das ist komfortabel, aber erfordert, dass Datenlieferanten Ihr Daten entsprechend bereitstellen.
Export (gepackt): Nicht unterstützt.
Export (dateibasiert): Nicht unterstützt.
Möglichkeit zur Gruppierung/Kollektionsbildung: Nicht unterstützt.
API: Keine API vorhanden.
Performance Import: Per OAI-PMH.
Performance Export: Export ist über VLO nicht möglich. Ein Zugriff via PURL ist abhängig vom referenzierten System, bei Tests war die Antwortzeit gut.
Performance Suche: Gut.
Clusterisierung: Nicht anwendbar. Es handelt sich um einen bereitgestellten Dienst.
Limit Anzahl der Objekte: Keine Einschränkungen bekannt.

[50] Single Sign-on (SSO), Ansatz zur Authentifizierung, bei dem sich ein Anwender nur einmal anmelden muss und im Rahmen einer Sitzung verschiedene Dienste nutzen kann.

Limit Speicher: Keine Einschränkungen bekannt. Da nur Metadaten gespeichert werden, sind die Anforderungen je Datensatz gering.
LZA Aspekte: Nicht bekannt.
Authentifizierung: Nicht notwendig, da ausschließlich gemeinfreie Daten gespeichert werden.
Autorisierung: Nicht unterstützt.
Link: https://vlo.clarin.eu
Besondere Eigenschaften:

- Verfügbarkeitscheck
- Prozessierung von verlinkten Ressourcen per "Language Resource Switchboard" Funktion, z.B. um Named Entity Recognition (NER) durchzuführen

5.1.3 DARIAH Collection Registry

Kurzbeschreibung: Web-Anwendung zur Bereitstellung von Informationen über Forschungssammlungen aus dem Bereich Kunst- und Geisteswissenschaften. Die Registry umfasst nur Beschreibungen, d.h. Metadaten und referenziert auf extern gespeicherte Ressourcen.
Metadaten: Begrenzter Umfang: Sprache, Titel, Beschreibung, Type der Sammlung, Lizenz und Rechte der Beschreibung und der referenzierten Ressource. DARIAH Collection Description Data Model (DCDDM)[51].
Identifikation: Per PID.
Versionierung: Die Beschreibungen der Sammlungen sind versioniert.
Zugriff: Die Beschreibung einer Sammlung ist nur über die Benutzungsschnittstelle zugreifbar. Der Zugriff auf die Ressource erfolgt per Referenz (PURL).
Suche: Über die Benutzungsschnittstelle. Wegen des begrenzten Umfangs der Metadaten ist die Suchfunktionalität limitiert.
Import (gepackt): Nicht unterstützt.
Import (dateibasiert): Eingabe der Metadaten über die Benutzungsschnittstelle.
Export (gepackt): Nicht unterstützt.
Export (dateibasiert): Nicht unterstützt.
Möglichkeit zur Gruppierung/Kollektionsbildung: Nicht unterstützt.
API: Keine API vorhanden.
Performance Import: Nicht anwendbar.
Performance Export: Nicht anwendbar.
Performance Suche: Gut.
Clusterisierung: Nicht anwendbar. Es handelt sich um einen bereitgestellten Dienst.
Limit Anzahl der Objekte: Keine Einschränkungen bekannt.
Limit Speicher: Keine Einschränkungen bekannt. Da nur Metadaten gespeichert werden, sind die Anforderungen je Datensatz überschaubar.
LZA Aspekte: Nicht bekannt.
Authentifizierung: Über die DARIAH Authentication and Authorization Infrastructure (AAI).
Autorisierung: Um neue Beschreibung anzulegen oder bestehende zu ändern ist eine Berechtigung erforderlich.
Link: https://colreg.de.dariah.eu/colreg-ui/
Besondere Eigenschaften:

[51] https://wiki.de.dariah.eu/pages/viewpage.action?pageId=52728662

- Alle Sammlungsbeschreibungen, die eine Referenz zu einer OAI-PMH-Schnittstelle enthalten, werden von der DARIAH-DE Generischen Suche indiziert.

5.1.4 DARIAH Repository

Kurzbeschreibung: Wie bei TextGrid Repository. Vereinfacht zusammengefasst fokussiert das TextGrid Repository auf Texte und zugehöriger Bilder (oder nur Bilder, sofern sie Texte darstellen), das DARIAH Repository (DH-Rep) ist für alles andere gedacht, also Geo-Daten, 3D-Modelle, etc.[52]
Link: https://de.dariah.eu/web/guest/repository

5.1.5 DataVerse

Kurzbeschreibung: Open-source Software zum Aufbau eines Forschungsdaten Repositoriums und einer Publikationensplattform für Forschungsdaten. In Dataverse können Datenbereiche (Dataverse) gebildet werden, die zur weiteren Strukturierung selbst wiederum Datenbereiche enthalten. Datenbereiche können zudem Datensätze, beschreibenden Metadaten, Dateien, die Dokumentation und Software, enthalten. Mit der Speicherung der Software, die zur Erstellung der Datensätze verwendet wurde, ist eine Arbeit nicht nur nachvollziehbar und überprüfbar, sondern auch reproduzierbar.
Metadaten: Die verwendeten Metadaten sind konfigurierbar. DataVerse unterscheidet zwischen Metadaten welche die Zitation unterstützen, den Datenbereich oder Datensatz beschreiben oder auf Dateiebene technische Metadaten bereitstellen. Es werden verschiedene Metadatenstandards[53] direkt unterstützt, z.B. Dublin-Core oder DataCite.
Identifikation: Über PID (DOI) für Dataverse, Dataset und Dateien.
Versionierung: Mit der Veröffentlichung eines Datenobjekts entsteht eine Version. Wenn Änderungen an Daten oder Metadaten vorgenommen werden, entsteht mit der Veröffentlichung der Änderungen eine neue Version.
Zugriff: Über die Benutzungsschnittstelle, oder REST API. Zugriff erfolgt immer über die PID.
Suche: Es werden nur begrenzte Metadaten bei der Suche berücksichtigt. Um eine umfangreichere Suche anbieten zu können, müssen Daten und Metadaten auf einen externen Suchindex abgebildet werden.
Import (gepackt): Nicht unterstützt. Entgegennahme und entpacken von ZIPs müsste von externen Dienst durchgeführt werden.
Import (dateibasiert): Über die Benutzungsschnittstelle und REST API.
Export (gepackt): Nicht unterstützt. DataVerse unterstützt den Download mehrerer Dateien mit einer Anfrage ("bundle" download), bei der eine PID Liste übergeben wird (sowohl über die Benutzungsschnittstelle, als auch REST API).
Export (dateibasiert): Über die Benutzungsschnittstelle oder REST API.
Möglichkeit zur Gruppierung/Kollektionsbildung: Gruppierung und Strukturierung über Datenbereiche möglich.
API: Ja. Umfangreiche und gut dokumentierte REST API.
Performance Import: Langsam bei großen Importaufträgen.
Performance Export: Gut.
Performance Suche: Gut.

---

[52] Abgrenzung TextGrid Repository und DARIAH Repository:
https://wiki.de.dariah.eu/display/publicde/Das+DARIAH-DE+Repository+und+das+TextGrid+Repository
[53] http://guides.dataverse.org/en/latest/user/dataset-management.html

Clusterisierung: Nicht bekannt.
Limit Anzahl der Objekte: Keine Einschränkungen bei eigener Installation, bei Nutzung eines bereitgestellten Dienstes, kann das anders sein.
Limit Speicher: DataVerse kann verschiedene Speichersysteme einbinden, z.B. S3, Dateisystem, Netzwerk Dateisystem, etc. Keine Einschränkungen bei eigener Installation, bei Nutzung eines bereitgestellten Dienstes, kann das anders sein.
LZA Aspekte: Die getestete Installation wird von der GWDG bereitgestellt.
Backup und Bitstream Preservation werden von der genutzten Infrastruktur behandelt. Metadaten der Langzeitarchivierung (z.B. Dokumentation von Datenmigration) werden vom System nicht gepflegt.
Authentifizierung: LDAP, SSO über Shibboleth.
Autorisierung: Rollenbasierte Zugriffskontrolle (Benutzer, Gruppen, Rollen)
Link: https://dataverse.org bzw. https://data.gro.uni-goettingen.de/
Besondere Eigenschaften:

- Bewertungsmaße, z.B. Anzahl der Zugriffe oder Downloads
- Veröffentlichte Datensätze können nicht gelöscht, sondern nur vor Zugriff geschützt werden
- umfangreiche Dokumentation

Hervorzuhebende Nachteile:

- Bei großen Importvorhaben dauert der Import lange

5.1.6 Fedora Commons

Kurzbeschreibung: Datenmanagement Software zum Aufbau eines Repositoriums. Mit der Software können beliebig strukturierte Daten gespeichert und mit Metadaten ausgezeichnet werden. Datenobjekte werden als Graph aufgebaut und können über Knoten strukturiert werden, an den Blätter können sich Dateien oder Metadaten befinden.
Metadaten: Zuweisung beliebiger Metadaten auf allen Ebenen möglich.
Identifikation: Knoten können über benutzerdefinierte PIDs adressiert werden. Wenn der Nutzer keine PID zuweist, erstellt das System selbst eine ID, so dass jeder Punkt Adressierbar ist.
Versionierung: Ja. Kann deaktiviert werden. Version des Gesamtobjekts muss jedoch explizit gesteuert werden.
Zugriff: Über REST API per ID des Knoten.
Suche: Die Suche wird über externe Systeme unterstützt. Sowohl ein Suchindex (Solr) und ein Triplestore (muss SPARQL unterstützen) kann integriert werden. Über den Triplestore können Beziehungen abgebildet werden (Linked Data), was eine Suche und die Navigation über Beziehungsgeflechte ermöglicht.
Import (gepackt): Nicht unterstützt. Entgegennahme und entpacken von ZIPs muss auf Applikationsebene durchgeführt werden.
Import (dateibasiert): Ein Graph wird nach und nach aufgebaut. Nachträgliche Ergänzungen (Dateien und Metadaten) sind möglich.
Export (gepackt): Nicht unterstützt. Muss auf Applikationsebene durchgeführt werden.
Export (dateibasiert): Über REST API, Knoten wird per URL adressiert, die mit der ID aufgebaut wird. Backup and Restore, ist jedoch kein Export im Sinne von OCR-D.
Möglichkeit zur Gruppierung/Kollektionsbildung: Beliebige Strukturierungsmöglichkeiten durch Beziehungen.
API: Umfangreiche und gut dokumentierte REST API.

Performance Import: Bei großen Importen bricht der Datendurchsatz ein.
Performance Export: Zugriff auf Dateien funktioniert gut, gebündelter Download aller Dateien eines Objekts ist nicht möglich und muss in Einzelschritten erfolgen.
Performance Suche: Gut über Suchindex und Triplestore.
Clusterisierung: Ist möglich.
Limit Anzahl der Objekte: Keine Einschränkungen bekannt.
Limit Speicher: Keine Einschränkungen bekannt. Ist abhängig vom eingebundenen Speichersystem (z.B. S3, NFS). Die Datenablage erfolgt nach Pair-Tree[54] Ansatz, womit eine Normalverteilung auf den Speichersystem erreicht wird.
LZA Aspekte: Nicht unterstützt. Backup und Bitstream Preservation müssen von der Infrastruktur behandelt werden. Metadaten der Langzeitarchivierung (z.B. Dokumentation von Datenmigration) müssen auf Applikationsebene gepflegt werden. Der Audit Service dokumentiert alle datenverändernden Aktivitäten (PROV-O und PREMIS).
Authentifizierung: Über Module (z.B. LDAP)
Autorisierung: Über Module. Rollenbasierte Zugriffskontrolle (Benutzer, Gruppen, Rollen)
Link: https://duraspace.org/fedora/
Besondere Eigenschaften:

- Modulares Konzept. Implementierung von Funktionen (Indexierung, Linked Data, Audit, etc.) über Dienste. Das Konzept kann für eigene Erweiterung genutzt werden.
- Graph-basiertes Datenmodell
    - Beliebige Strukturierungsmöglichkeiten
    - Jeder Knoten im Graphen ist adressierbar
    - Direkte Unterstützung für Linked-Data
    - Erlaubt komplexe Suchen
    - Integration eines Triplestore
- Prüfsummencheck nach Übergabe
- Referenzierung extern gespeicherter Ressourcen, Fedora Commons kann damit auch ausschließlich für die Metadatenverwaltung eingesetzt werden
- Unterstützt Transaktionen
- Audit Service, der Aktivitäten innerhalb des Systems sammelt und Provenienzdaten im PROV-O und PREMIS Format dokumentiert (z.B. Ingest, Update).
- Metadaten auf allen Ebenen
- Gute Dokumentation

Hervorzuhebende Nachteile:

- Versionierung des Gesamtobjekts muss explizit gesteuert werden.
- Export nur über Einzeldateien.
- Mit eingebundenem Index und Triplestore dauerte der Import bei großen Importvorhaben lange, ohne Index und Triplestore, war der Import schnell.
- Updates in der Vergangenheit haben mit dem Datenmodell und der Schnittstelle gebrochen. Es wurden zwar immer Migrationsphase angeboten, aber der Aufwand der Datenmigration blieb erheblich.

[54] Mit Pair-Tree wird ein Ausdruck in 2-er Blöcke aufgeteilt. Bei der Ablage in einem Dateisystem wird damit eine bessere Verteilung auf der Festplatte erreicht.

5.1.7 GitHub

Kurzbeschreibung: System zur Versionsverwaltung und das Source-Code-Management in Softwareprojekten.
Metadaten: Name, Beschreibung, Organisation, Besitzer, Sichtbarkeit (privat, öffentlich), Lizenz. Texte werden volltext-indexiert.
Identifikation: Über Name der Organisation und des Nutzers und des Repositoriums. Keine PID/PURL.
Versionierung: Implizit. Jeder Commit erzeugt eine Version. Bestimmte Commits können mit einer Marke (Tag) versehen werden, so dass der Zustand als besondere Version (Meilenstein, Release) hervorgehoben und benannt wird.
Zugriff: Über Kommandozeilenwerkzeug (CLI) oder andere Werkzeuge (z.B. Entwicklungsumgebung), z.B. Download der gesamten Versionsgeschichte (fetch), Download der Änderungen im aktuellen Branch (pull).
Suche: Gut für die Suche innerhalb eines Repositoriums. Suche über alle Repositorien im GitHub über Namen von Repositorien, Beschreibungen und Volltext-indexierte Textbasierte Inhalte ist nicht komfortabel.
Import (gepackt): Nicht unterstützt. Entgegennahme und entpacken von ZIPs müsste von externen Dienst durchgeführt werden.
Import (dateibasiert): Initialer Commit (push) bildet die Basis. Bei nachträglichen Änderungen (neue, geänderte oder gelöschte Objekte) werden nur die Änderungen gespeichert.
Export (gepackt): Nicht unterstützt (im Sinne von OLA-HD). ZIPs könnten von externem Dienst erzeugt werden. Für Archivierungszwecke ist es mit dem archive Kommando möglich den aktuellen Zustand des GitHub Repositoriums als ZIP zu exportieren. Es handelt sich dabei aber nicht um einen Export wie er in OCR-D erwartet wird, da nicht nur die Nutzerdaten exportiert werden, sondern auch alle Daten- und Metadaten der Versionsverwaltung.
Export (dateibasiert): Es kann per URL gezielt auf eine Datei (raw content) zugegriffen werden.
Möglichkeit zur Gruppierung/Kollektionsbildung: Über Subprojekte ist eine Strukturierung möglich, wobei das schnell unübersichtlich wird.
API: Zugriff via CLI[55].
Performance Import: Upload umfangreicher Commits dauert lange, da versucht wird, Konflikte (wenn ein vorausgegangener Commit, dieselben Stellen bearbeitet hat) aufzulösen. Ggf. ist eine manuelle Lösung von Konflikten erforderlich.
Performance Export: Gut.
Performance Suche: Gute Suche im Repositorium, aber unkomfortable Suche über alle GitHub Repositorien.
Clusterisierung: Nicht unterstützt. Es handelt sich um einen bereitgestellten Dienst.
Limit Anzahl der Objekte: GitHub verwaltet viele Millionen Softwareprojekte, aber es ist wegen fehlender Strukturierungsmöglichkeiten nicht die passende Umgebung für OCR-D oder die massenhafte Digitalisierung.
Limit Speicher: Die kostenfreie Nutzung ist beschränkt (bzgl. Anzahl privater Projekte und Speicherplatz). Es gibt unterschiedliche Kostenpläne.
LZA Aspekte: Nicht explizit unterstützt. Backup wird über die Infrastruktur sichergestellt. Zudem handelt es sich um eine verteilte Versionskontrolle. Mit jeder Kopie wird das Repositorium abgesichert.

[55] Command Line Interface. Programm das in der Console (bzw. Terminal oder CMD-Fenster) ausgeführt wird.

Authentifizierung: Nutzerkonto mit Basic Authentication (user/password) oder 2-Faktor-Authentifizierung[56].
Autorisierung: Sichtbarkeit von privaten Repositorium nur für autorisierte Personen und Gruppen.
Link: https://github.com
Besondere Eigenschaften:

- Erzeugung von Entwicklungszweigen (branches)
- Zusammenführen von Entwicklungszweigen
- Vollständige Abbildung der Entwicklungsgeschichte
- Zusätzliche Datensicherheit durch Kopien
- Gute Unterstützung für verteilte Arbeit (Vorbereitung der Archivierung)

Hervorzuhebende Nachteile:

- Ungeeignet zur Ablage großer Datenbestände.
- Kosten

5.1.8 Internet Archive

Kurzbeschreibung: Das Internet Archive ist ein offenes Archiv, u.a. für digitalisierte Bücher, Audiodateien, Videos, Bilder oder archivierte Internetseiten und Software.
Metadaten: MARCXML oder MARC21
Identifikation: Open Library ID, Internet Archive ID, OCLC/WorldCat ID
Versionierung: Über die Benutzungsschnittstelle oder per API
Zugriff: Über die Benutzungsschnittstelle. Anzeige der Metadaten oder Anzeige von Buchseiten in einem integrierten Image Viewer. Zugriff per API.
Suche: Gute Suche und Navigation über die Benutzungsschnittstelle. Suche per API.
Import (gepackt): ZIP-basierter Import mit Bildern und optionalen Metadaten im MARC Format (MARCXML, MARC21). Kein ZIP Import im Sinne von OCR-D.
Import (dateibasiert): Nur einzelne Dateien.
Export (gepackt): Nicht unterstützt.
Export (dateibasiert): Nur einzelne Dateien.
Möglichkeit zur Gruppierung/Kollektionsbildung: Möglich. Anfrage per E-Mail erforderlich.
API: Eine an S3 angelehnte REST API (Bucket, Key), Python Bibliothek.
Performance Import: Nicht gut. Aufwändig, da Einzeldateien importiert werden.
Performance Export: Gut.
Performance Suche: Gut.
Clusterisierung: Nicht unterstützt. Beim Internet Archive handelt es sich um einen bereitgestellten Dienst.
Limit Anzahl der Objekte: Nicht bekannt.
Limit Speicher: Nicht bekannt.
LZA Aspekte: Nicht bekannt.
Authentifizierung: Basic Authentication (user/password).
Autorisierung: Import erfordert Anmeldung am System.
Link: archive.org
Besondere Eigenschaften:

- OCR Erzeugung für importierte Textbasierte Dokumente

[56] Authentifizierung erfolgt durch zwei unabhängige Sicherungskomponenten.

Hervorzuhebende Nachteile:

- Import von Einzeldateien oder im ZIP Dateiformat mit Seiten eines Buchs, aber keine ergänzenden Daten/Metadaten, Datenmodell erscheint ungeeignet für OCR-D

5.1.9 TextGrid Repository

Kurzbeschreibung: Das TextGrid Repository ist ein Langzeitarchiv für geisteswissenschaftliche Forschungsdaten. Es speichert Texte und Bilder, unterstützt die Suche und den Zugriff, sowie die dauerhaft stabile Zitation publizierter Forschungsdaten.
Metadaten: Werden mit dem TG Metadaten Schema festgelegt[57].
Identifikation: Über TextGrid URIs und über Handle PIDs. Die API integriert den GWDG PID Service, der Handle IDs erzeugt und registriert. Die Auflösung der TextGrid URIs erfolgt über den TextGrid Resolver Service.
Versionierung: Über TG-crud (REST API). Ein erneuter Import erzeugt eine neue Objektversion (Revision).
Zugriff: Über TG-crud (REST API)
Suche: Über TG-search (REST API). Suchausdrücke können mit logische Operatoren und Funktionen der zugrundeliegenden Suchmaschine (Lucene) aufgebaut werden, um komplexe Suchterme zu formuliert. Die Suche unterstützt Paginierung.
Import (gepackt): Nein. Importpakete müssen konform zu der im TG Metadaten Schema beschriebenen Struktur sein und die geforderten Metadaten enthalten. Für Primärdaten (ausschließlich) ist ein Import durch Übergabe von METS/MODS oder IIIF Beschreibungen möglich. Aus den Dokumenten werden Metadaten extrahiert und die Dokumente (Bilder, Volltexte) werden von den referenzierten Stellen geladen.
Import (dateibasiert): Über TG-crud (REST API) und TG-import (Applikation).
Export (gepackt): Nein. TextGrid Aggregator (REST API) erlaubt den Export abgeleiteter Dokumente, wobei z.B. HTML, ePub oder PDF Repräsentationen erzeugt und zurückgegeben werden.
Export (dateibasiert): Über TG-crud (REST API).
Möglichkeit zur Gruppierung/Kollektionsbildung: Daten in einem TextGrid-Projekt können über Kollektionen, Editionen und Aggregationen strukturiert werden.
API: Umfangreich und über Module gut strukturierte REST APIs. Java-basierter Import mit TG-import (koLibRI). Manueller Import über das TextGridLab[58].
Performance Import: Nicht gut, d.h. aufwändig, da Einzeldateien importiert werden. Import via METS/MODS oder IIIF ist komfortabler, aber nicht performant.
Performance Export: Gut.
Performance Suche: Gut über Suchindex und Triplestore.
Clusterisierung: Externer Dienst.
Limit Anzahl der Objekte: Kein Limit.
Limit Speicher: Kein Limit.
LZA Aspekte: Backup und Bitstream Preservation werden von der genutzten Infrastruktur behandelt. Provenienzdaten und Metadaten der Langzeitarchivierung (z.B. Dokumentation von Datenmigration) werden vom System nicht gepflegt.
Authentifizierung: SSO über Shibboleth (über DFN-AAI: mit DARIAH-Account oder Föderationsaccount).

---

[57] https://wiki.de.dariah.eu/display/TextGrid/Metadaten-Editor-Sicht
[58] https://textgridlab.org/doc/)

Autorisierung: Über TG-auth per RBAC (rollenbasierte Zugriffskontrolle) unterstützt.
Link: https://textgrid.de/, https://textgridrep.org/
Besondere Eigenschaften:

- Export abgeleiteter Inhalte (z.B. PDF) per TG Aggregator
- METS und IIIF Import
- OAI-PMH Server zur Bereitstellung von Metadaten
- Integration eines Triplestore
- Umfangreiche und gut nutzbare Dokumentation
- Zertifiziert mit dem Core Trust Seal (CTS) (https://www.coretrustseal.org/why-certification/certified-repositories/)

5.1.10 Zenodo

Kurzbeschreibung: Open-access Repositorium für forschungsbezogene digitale Artefakte, z.B. Datensätze, Forschungssoftware, Berichte usw.
Metadaten: Festgeschriebener Satz an Metadaten[59].
Identifikation: Für jedes importierte Objekt wird ein eindeutiger und persistenter Digital Object Identifier (DOI) erzeugt und registriert.
Versionierung: Über REST API[60] unterstützt.
Zugriff: Über Benutzungsschnittstelle oder REST API[61].
Suche: Über Benutzungsschnittstelle oder REST API[62].
Import (gepackt): Nicht unterstützt. Entgegennahme und entpacken von ZIPs müsste von einem externen Dienst durchgeführt werden.
Import (dateibasiert): Über Benutzungsschnittstelle oder REST API[63].
Export (gepackt): Nicht unterstützt. Müsste von einem externen Dienst durchgeführt werden.
Export (dateibasiert): Über Benutzungsschnittstelle oder REST API[64].
Möglichkeit zur Gruppierung/Kollektionsbildung: Nicht unterstützt.
API: Umfangreiche und gut strukturierte REST APIs.
Performance Import: Gut (aber Einzeldateien).
Performance Export: Gut (Zugriff auf Dateien).
Performance Suche: Gut.
Clusterisierung: Nicht anwendbar. Bei Zenodo handelt es sich um einen bereitgestellten Dienst.
Limit Anzahl der Objekte: Nicht bekannt.
Limit Speicher: Import ist auf 50 GB pro Datensatz begrenzt.
LZA Aspekte: Nicht bekannt.
Authentifizierung: Basic Authentication (user/password)[65]
Autorisierung: Autorisierung der REST-Aufrufe per Zugriffstoken. Es gibt unterschiedliche Autorisierungsebenen (scopes: create/publish, edit, discard)

---

[59] https://developers.zenodo.org/
[60] https://developers.zenodo.org/#new-version
[61] https://developers.zenodo.org/#retrieve
[62] https://developers.zenodo.org/#records
[63] https://developers.zenodo.org/#create
[64] https://developers.zenodo.org/#retrieve
[65] https://developers.zenodo.org/#authentication

Link: https://zenodo.org
Besondere Eigenschaften:

- Zitierbarkeit/Referenzierbarkeit per Digital Object Identifier (DOI)
- OAI-PMH Server zur Bereitstellung von Metadaten. Unterstützte Formate sind z.B. oai_dc, marc21 oder oai_datacite
- Bewertungsmaße, z.B. Anzahl der Zugriffe oder Downloads nach COUNTER

5.2 Beurteilung

Mit der Klärung der technischen Anforderungen zu Beginn des Projektes wurden Kriterien für die Systemauswahl bzw. eine kombinierte Systemkonfiguration festgelegt. Die untersuchten Systeme archivematica, CDStar, CLARIN Virtual Language Observatory (VLO), DARIAH Collection Registry, DARIAH Repository, DataVerse, Fedora Commons, GitHub, Internet Archive, TextGrid Repository und Zenodo adressieren unterschiedliche Anwendungsbereiche. Der Funktionsumfang reicht von Metadatenmanagement, Speichersystem und Versionsverwaltung, bis zur integrierten Lösung, die alle Aspekte zusammenzubringen versuchen.

Reine Metadatenmanagement-Lösungen, wie die DARIAH Collection Registry oder CLARIN Virtual Language Observatory (VLO), speichern keine Daten, sondern referenzieren auf andere Systeme. Sie erfüllen, für sich genommen, nicht die gestellten Anforderungen. Dennoch können sie in Kombination mit einer Datenmanagement Lösung, bspw. CDSTAR, die Funktionen des Archivs, um Funktionalitäten (hier die Recherche), ergänzen. Trotz der Einschränkungen bieten diese Systeme interessante Funktionen. So ist es bspw. mit VLO per “Language Resource Switchboard" möglich verlinkten Ressourcen zu laden und Prozessierungen darauf auszuführen, z.B. um Named Entity Recognition (NER) durchzuführen.

Die Systemevaluierung hat ergeben, dass keines der Systeme den erwarteten Funktionsumfang allein erfüllt. Insbesondere die OCR-D-typischen Anforderungen bezüglich der Importpakete (OCRD-ZIP) und Langzeitarchivierung sind unzureichend gelöst. Das System archivematica erschien zunächst als die gesuchte Lösung. Es handelt sich dabei um ein webbasiertes System zur Archivierung, das nach dem OAIS Referenzmodell aufgebaut ist. Ein Import ist im ZIP Format möglich. Es unterstützt aber keine Versionierung, eine Trennung von Produktiv- und Archivspeicher ist nicht möglich und es fehlt eine Programmierschnittstelle, so dass für jeden Import nach der Datenübernahme eine manuelle Konfiguration des AIP über die Benutzerschnittstelle durchgeführt werden muss, was bei großen Importvorhaben nicht durchführbar ist. Auch das unter Beteiligung der SUB Göttingen entwickelte TextGrid Repository oder die Datenmanagement Software Fedora Commons bieten bereits viele geforderte Funktionen. Bei genauerer Betrachtung wurde jedoch deutlich, dass auch diese Systeme für den geplanten Einsatzbereich ungeeignet waren, da bspw. geforderte Import- oder Exportformate, die Trennung von Produktiv- und Archivspeicher, oder benötigte Schnittstellen (z.B. gepackter Import und Export) nicht oder nicht angemessen unterstützt werden. Fedora Commons unterstützt zwar umfangreiche Möglichkeiten zur Versionierung und Strukturierung, die Versionierung des Gesamtobjekt muss aber explizit gesteuert werden und erscheint nicht durchdacht, zudem war der Import sehr langsam. Weder Fedora Commons noch TextGrid Repository Unterstützen die Trennung von Produktiv- und Archivspeicher, es wäre ein weiteres System zur Einbindung von Bandspeicher erforderlich,

um Archivobjekte mit vertretbaren Kosten langfristig aufzubewahren zu können. Wenn die Einbindung nicht unmittelbar unterstützt wird, dann können Inkonsistenzen entstehen. Um Langzeitarchivierungsaspekte gut unterstützen zu können, ist ein System erforderlich, das Daten exakt im Einlieferungszustand speichert, aber zugleich die Adressierung und den Zugriff auf Archivobjekte und deren Bestandteile ermöglicht.

Es wurde sich infolge der Evaluierung für eine Implementierung entschieden, welche auf verschiedene technische Komponenten aufbaut (s. folgendes Kapitel). Die Implementierung ermöglicht eine lose gekoppelte Architektur aus Einzelkomponenten, die je nach Nutzungsszenario ausgetauscht werden können. Dies erhöht die Flexibilität, die Nachnutzungsmöglichkeiten und senkt die Kosten des Gesamtsystems. Allerdings stellt dies gleichzeitig eine Herausforderung bei der Portabilität des Gesamtsystems dar, da es mehrere Container-Virtualisierungen (z.B. Docker-Instanzen) oder laufende Services (wie. z. B. Verzeichnisdienst oder Bitstream-Preservation) erfordert.

6 OLA-HD - Der Prototyp

Beim OLA-HD, das im Rahmen des Projekts prototypisch entwickelt wurde, handelt es sich um eine Webanwendung, die eine web-basierte Benutzungsschnittstelle und eine REST-basierte Programmierschnittstelle (API) anbietet. Der Produktiv- und Archivspeicher wird über den Archiv-Manager (s. 2.6 Archiv-Manager CDSTAR) eingebunden. Administrative Metadaten speichert der Archiv-Manager in einer dokumentenorientierten Datenbank (MongoDB[66]), beschreibende Metadaten und Volltexte werden in einer Suchmaschine (ElasticSearch[67]) indiziert. Das Nutzermanagement und die Erzeugung von PIDs, sowie deren Auflösung, erfolgt durch externe Dienste (s. 3.2.6 Identifikation von Archivobjekten). Technische Aspekte der Langzeitsicherung, wie Datensicherung, Bitstream Preservation oder Datenmigrationen auf der Ebene der Speichersysteme, werden von der IT Infrastruktur sichergestellt.

Mit dem OLA-HD-Prototypen[68] kann der Nutzer – Mensch oder Maschine – OCR-Ergebnisse eines Werkes im OCR-D-Archivformat (OCRD-ZIP) in das System laden. Das System validiert die Datei, ob sie OCRD-ZIP konform ist und ob geforderte Metadaten enthalten sind, bezieht und registriert eine PID und weist sie dem Archivobjekt (der ZIP Datei) zu und schickt die Datei an den Archiv-Manager. Dieser schreibt die Datei in das Archiv (Bandspeicher). Zudem wird das Importpaket entpackt, Daten werden für die Suche indexiert und Abhängig von der Konfiguration (Datei-Typ, Datei-Größe etc.) werden Dateien zusätzlich in einem Produktivspeicher gespeichert, um einen schnellen Zugriff zu ermöglichen. Auch dem Produktivspeicherobjekt wird eine eigene PID zugewiesen. Der Nutzer hat Zugriff auf alle OCR-Versionen und kann diese zudem als OCRD-ZIP Dateien exportieren und herunterladen. Alle Werke und Versionen haben eigene PIDs. Die PIDs werden vom European Persistent Identifier Consortium (ePIC) [69] Service generiert. Die Entscheidung für ePIC PIDs und gegen beispielsweise DataCite DOIs begründet sich darin, dass DOIs für (Daten-)Publikationen bestimmt sind und ePIC eine freie Wahl der Metadaten erlaubt. Die verschiedenen OCR-

[66] https://www.mongodb.com

[67] https://www.elastic.co/de/enterprise-search

[68] https://github.com/subugoe/OLA-HD-IMPL

[69] https://www.pidconsortium.net/

Versionen eines Werkes sind über die PID verknüpft, so dass das Datenmanagementsystem die Versionierung in einer Baumstruktur abbilden kann. Durch Vergabe persistenter Identifier in Kombination mit der Speicherung von Modifikationszeitpunkten sind Zustände eines Werkes zu einem bestimmten Zeitpunkt adressierbar und zitierbar.

Abgesehen von dem Hochladen von Daten und dem Herunterladen von hochaufgelösten Bildern können Gastbenutzer das System in vollem Umfang nutzen. Sie können beispielsweise das Repositorium durchsuchen und eine Vorschau von Text und – falls verfügbar – Bildern in der Dateistruktur erhalten oder durch die verschiedenen Versionen navigieren. Nutzer können sich registrieren und ihre Daten in einem Dashboard verwalten.

Der prototypische Charakter des OLA-HD – evident und kalkuliert im Rahmen der relativ kurzen Projektlaufzeit – zeigt sich insbesondere durch Kompromisse bei der Usability. Die Benutzungsschnittstelle, Trefferlisten und Detailansichten können für ein Produktivsystem überarbeitet und ansprechender gestaltet werden. Die Möglichkeiten der Suche, wie z. B. Filterung, werden bislang nicht genutzt. Die prototypische Entwicklung liefert eine solide Basis für ein perspektivisches Usability Engineering, um den Funktionsumfang sinnvoll zu erweitern.

Die prototypische Umsetzung wurde in einer Virtualisierungsumgebung (als Docker-Instanz) bereitgestellt, um weitere Evaluationen und Nachnutzungen zu ermöglichen. Die Softwarekomponenten wurden in Github[70] quelloffen zur Verfügung gestellt. Die entwickelte Software wurde unter der Apache Software License 2.0 und das schriftliche Konzept[71] sowie die Dokumentationen unter CC-BY-SA 4.0 zur Verfügung gestellt, um die freie Nachnutzbarkeit zu gewährleisten. Auf der OLA-HD Website[72] (Demo-Seite) kann das System getestet werden.

Ein Packaging-Werkzeug ermöglicht die einfache Erzeugung von Importpaketen und deren Übergabe an die Importschnittstelle des Archivs. Das Werkzeug analysiert die METS Strukturen, lädt die referenzierten Dokumente (z.B. Bilder in unterschiedlichen Auflösungen, Volltexte, etc.), speichert sie lokal gemäß OCRD-METS, korrigiert Referenzen im METS (FileSec), packt das Importpaket gemäß OCRD-ZIP und startet die Übergabe an die OLA-HD Importschnittstelle. Das Werkzeug kann als Demonstrator für die Integration dienen und in Digitalisierungsworkflows eingebunden werden.

6.1 Implementierung

Die Implementierung der Benutzungsschnittstelle des OLA-HD wurde mit dem Java-Script Framework VueJS[73] und dem Frontend-CSS-Framework Bootstrap[74] implementiert, die REST API basiert auf Spring Boot[75]. Die Implementierung der Programmierschnittstelle wird in einen Tomcat Applikationsserver installiert. Der Source-Code ist im GitHub Repositorium

[70] https://github.com/subugoe/OLA-HD-IMPL

[71] https://github.com/subugoe/OLA-HD-IMPL/raw/master/docs/OLA-HD_Konzept.pdf

[72] http://ola-hd.gwdg.de, http://ola-hd.sub.uni-goettingen.de

[73] https://vuejs.org

[74] https://getbootstrap.com

[75] https://spring.io/projects/spring-boot

subugoe/OLA-HD-IMPL[76] abgelegt. Für eine einfache Installation wird Docker[77] und Docker-Compose[78] verwendet. Der Archiv-Manager ist nicht Teil der Docker-Konfiguration und muss separat installiert werden.

6.1 Technische Umgebung

Die Anwendung wurde in einer GWDG Cloud Server VM installiert. Die Server Konfiguration umfasst 4 Kerne, 8 GB RAM und 80 GB HDD. Im Testbetrieb war die Konfiguration ausreichend und stieß nie an ihre Grenzen.

Es muss jedoch angemerkt werden, dass bei der Konfiguration des Prototyps, auf eine bestehende Installation des Archiv-Managers zurückgegriffen wurde. Diese Komponente beinhaltet ressourcenintensiven Funktionen, so dass die Serverkonfiguration entsprechend erhöht werden muss, wenn der Archiv-Manager lokal installiert werden soll (16 Kerne, 32 GB RAM und 160 GB HDD).

6.3 Architektur

Die Gesamtarchitektur kombiniert und integriert existierende Softwarekomponenten und bestehende Systeme. Es wird ein modulares Konzept verfolgt, bei dem einzelne Komponenten austauschbar sind. So können der Suchindex oder der Archiv-Manager sowohl als Teil einer Docker-Umgebung, als auch als eigenständige Dienste betrieben werden. Der Archiv-Manager kann durch eine andere Datenmanagement- und Speicherlösung ersetzt werden. Die lose Kopplung der Komponenten über die Programmierschnittstelle, ermöglicht auch die Integration der Benutzungsschnittstelle in bestehende Systeme. Das Nutzer- und Berechtigungsmanagement wird aktuell durch einen Verzeichnisdienst (Active Directory) ausgeführt. Es kann durch andere Lösungen ersetzt werden, die das LDAP Protokoll unterstützen. Um die Komplexität zu reduzieren, ist der Archiv-Manager nicht Bestandteil der Docker-Umgebung und wird als Dienst der GWDG eingebunden, und zudem nicht bekannt ist, ob die Zielumgebung bspw. Zugriff auf einen Archivspeicher hat.

[76] https://github.com/subugoe/OLA-HD-IMPL

[77] https://www.docker.com

[78] https://docs.docker.com/compose/

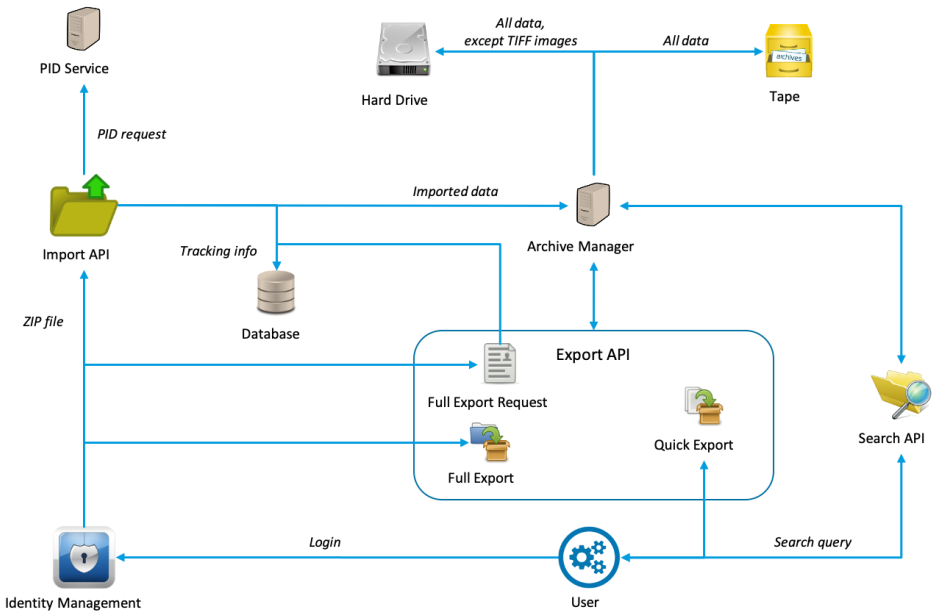

Abb. 1: Architektur des OLA-HD Prototyp

6.4 Interaktion mit dem Archiv

6.4.1 Import

Beim Import muss sich der Nutzer am System anmelden. Das System unterstützt sowohl eine einfache Authentifizierung per Benutzername und Passwort, als auch einen token-basierten Ansatz. Beim token-basierten Ansatz, wird ein geschützter Kontext geschaffen, in dem z.B. die geschützte API Funktion des Imports ausgeführt werden kann. Mit dem Token wird die Autorisierung des Aufrufs von geschützten Funktionen nachgewiesen.

Nach Übernahme des Importpaketes überprüft der Dienst, der sich hinter der Importschnittstelle verbirgt, zunächst die korrekte Datenübernahme (Prüfung der Prüfsummen), die Konformität zu OCRD-ZIP und ob alle erforderlichen Metadaten enthalten sind. Wenn Fehler festgestellt werden, dann bekommt der Nutzer eine Fehlermeldung und muss den Import, ggf. mit korrigiertem Importpaket, erneut ausführen.

Wenn bei den vorangegangenen Überprüfungen des Importpaketes nichts zu beanstanden war, wird für das Produktiv- und Archivspeicherobjekt je eine eigene PID bezogen und registriert. Danach übergibt die Importschnittstelle das Importpaket an den Archiv-Manager. Der Archiv-Manager speichert das unveränderte Importpaket im Archivspeicher. Zudem entpackt der Archiv-Manager das Importpaket und speichert Teile, die gemäß Konfiguration für den direkten Zugriff zur Verfügung stehen sollen, im Produktivspeicher. Beschreibenden Metadaten werden auf einen Suchindex abgebildet, text-basierte Daten werden volltext-indexiert.

Informationen über das Importpaket (z.B. Größe, Prüfsumme), den Importvorgang (z.B. Datum, Status des Importes), sowie administrative Metadaten (z.B. Besitzer, Konfiguration) werden in einer Datenbank gespeichert.

Wenn die Datenübernahme abgeschlossen ist oder wegen aufgetretener Fehler abbricht, wird der Status des Imports aktualisiert (von processing auf success oder failed). Im Fehlerfall werden bereits ausgeführte Aktionen wieder zurückgenommen, damit es nicht zu Inkonsistenzen kommt.

6.4.2 Export aus dem Produktivspeicher (Quick Export)

Der Export kann mit einer Suche im Archiv beginnen. Wenn die PID oder der Identifier (Record Identifier) des gewünschten Archivobjekts bekannt ist, dann kann damit der direkte Zugriff auf das Archivobjekt im Produktivspeicher über die Export API durchgeführt werden. Der Dienst hinter der Export API leitet die Anfrage an den Archiv-Manager weiter, der das Archivobjekt zurückgibt und damit den Download veranlasst.

6.4.3 Export aus dem Archivspeicher (Full Export Request & Full Export)

Der Zugriff auf den Archivspeicher ist geschützt, weil durch Missbrauch das Archivspeichersystem gestört oder sogar lahmgelegt werden kann. Der Nutzer muss sich wie beim Import am System anmelden, um einen Zugriffstoken zu erhalten.

Da der Zugriff auf Archivobjekte im Archivspeicher zeitaufwändig ist und u.U. Stunden oder Tage in Anspruch nehmen kann, wurde ein zweistufiger Zugriff implementiert. Im ersten Schritt stellt die Export API beim Archiv-Manager eine Anfrage, so dass dieser das geforderte Archivobjekt (per PID oder Identifier) bereitstellen kann. Der Archiv-Manager lädt daraufhin das Archivobjekt aus dem Archivspeicher und speichert es vorübergehend (zeitlich befristet, oder bis Zugriff erfolgt ist) auf der lokalen Festplatte. In einem zweiten Schritt kann der Nutzer den Zugriff auf das bereitgestellte Archivobjekt (wieder per PID oder Identifier) durchführen. Wenn die Bereitstellung bei Zugriff noch nicht abgeschlossen ist, dann bekommt der Nutzer eine Fehlermeldung und der Zugriff muss zu einem späteren Zeitpunkt wiederholt werden.

6.4.4 Suche

Die Such API ist für die Interaktion über eine Präsentationsschicht gedacht. Das OLA-HD hat über die verschiedenen Sichten hinweg, immer ein Textfeld eingebunden, über das der Nutzer Suchterme eingeben und Suchanfragen absenden kann. Der Dienst hinter der Such API nimmt den Suchstring entgegen, baut daraus eine Anfrage, die der Suchindex versteht, ggf. nimmt er auch Änderungen am Suchstring vor (z.B. entfernen oder schützen von Sonderzeichen), und sendet die Anfrage dann an den Suchindex. Der Dienst erzeugt aus dem Suchergebnis eine anzeigbare Repräsentation der Daten.

Der Such API kann einen Suchterm oder ein ElasticSearch Suchausdruck übergeben werden. Mit dem Suchausdruck kann gezielt über Metadatenfeldern gesucht werden und es können logische Verknüpfungen (z.B. UND, ODER) in Suchausdrücken formuliert werden. Die Trefferliste ist standardmäßig auf 25 Treffer limitiert, der Wert kann bei einer Anfrage angepasst werden. Zudem wird mit der Trefferliste eine Paginierungs-ID zurückgegeben, so dass über große Trefferlisten navigiert werden kann.

Der Zugriff auf ein Archivobjekt kann per PID oder Identifier, die Teil der Metadaten der Treffer sind, über die Export API erfolgen oder direkt über die Benutzungsschnittstelle.

6.4.5 Zugriff über Benutzungsschnittstelle

Die Suche kann auch über die Benutzungsschnittstelle durchgeführt werden. Durch Eingabe des Suchterms in das Textfeld für die Suche und Absende einer Suchanfrage, wird die Such API aufgerufen. Als Antwort wird eine Trefferliste zurückgegeben.

Nach Auswahl eines Archivobjekts aus der Trefferliste besteht in einer Detailansicht die Möglichkeit, Bestandteile des Archivobjekts anzuzeigen (durch folgen der angezeigten Verknüpfung), um den Text einer Seite zu lesen und beurteilen zu können, ob es sich bei dem ausgewählten Archivobjekt um das richtige handelt. Zudem können einzelne Dateien, oder ganze Verzeichnisbäume über eine Auswahlkomponente ausgewählt werden, um sie für den Export zu markieren, und anschließend einen Export als ZIP Archiv zu veranlassen.

6.4.6 Legende

User: Ein Dienst bei der REST-basierten Maschine-zu-Maschine Interaktion, ein menschlicher Nutzer bei der Benutzungsschnittstelle.

Identity management: Aktuell wird das Nutzermanagement über den GWDG OpenLDAP Dienst durchgeführt. Um Daten in das OLA-HD importieren oder eine vollständige Kopie aus dem Archivspeicher laden zu können, müssen sich Nutzer im System anmelden.

Import API: Ein REST Endpunkt, an dem Nutzer Importpakete an das System übergeben können. Diese Komponente ruft den PID Service auf, um einen neuen Persistent Identifier (PID) für die Produktiv- und Archivspeicherobjekte zu erhalten und zu registrieren, sie speichert ergänzende Informationen in einer Datenbank (z.B. Besitzer des Archivobjekts, Importzeit), Indexiert Volltexte und schickt die Daten anschließend zum Archive Manager Dienst.

PID Service: Es wird der GWDG PID Service für die Erzeugung von PIDs verwendet, der Handle IDs erzeugt. Für die Auflösung der PIDs wird der Dienst Handle.Net eingesetzt, der eine Weiterleitung auf das Archivobjekt vornimmt.

Database: Die Datenbank des OLA-HD, die alle wichtigen Informationen zu den Archivobjekten speichert.

Archive manager: Ein Dienst (CDSTAR) der die Kommunikation mit den persistenten Speicherungssystemen (Produktiv- und Archivspeicher) steuert. Ein neu importiertes Archivobjekt überträgt der Dienst vollständig, d.h. unverändert, an den Archivspeicher. Um einen unmittelbaren, d.h. schnellen, Zugriff zu ermöglichen, wird ein Teil der Daten im Produktivspeicher abgelegt. In der aktuellen Konfiguration, werden hochauflösende TIFF Bilder nicht im Produktivspeicher abgelegt.

Export API: Über die Export API kann auf Archivobjekte zugegriffen werden.

- **export**: Dieser Endpunkt erlaubt den direkten Zugriff auf Archivobjekte im Produktivspeicher. Eine Anmeldung ist nicht erforderlich.
- **full-export**: Hierüber kann nach vorausgegangenem Export Request auf Objekte im Archivspeicher zugegriffen werden. Der Zugriff auf den Archivspeicher ist geschützt.
- **export request**: Um auf Daten im Archivspeicher zuzugreifen, müssen diese zunächst aus dem Archivspeicher geladen und auf Festplatte gespeichert werden. Da dies nicht unmittelbar durchgeführt werden kann und die Bereitstellung u.U. Stunden oder Tage in Anspruch nimmt, erfolgt der Zugriff auf den Archivspeicher in zwei Schritten. In einem ersten Schritt wird der Zugriff angemeldet und der Archive Manager veranlasst das laden aus dem Archivspeicher. In einem Zweiten Schritt kann dann versucht werden, auf das bereitgestellte Archivobjekt zuzugreifen. Wenn die Vorbereitungen noch nicht abgeschlossen sind, dann erhält der Nutzer eine Fehlermeldung und kann zu einem späteren Zeitpunkt erneut versuchen zuzugreifen. Die Bereitstellung ist nur vorübergehend, und wird automatisch wieder gelöscht. Die Bereitstellungsdauer ist Teil der Konfiguration.

Search API: Menschliche Nutzer können über die Benutzungsschnittstelle eine Suche über Metadaten und Volltexte durchführen.

6.5 API Beschreibung

Die OLA-HD API umfasst Funktionen zum

- Import und Export von Archivobjekten,
- Anmelden am System, um einen token-basierten geschützten Kontext zu schaffen, in dem geschützte Funktionen aufgerufen werden können, z.B. Import eines neuen Archivobjekts oder Zugriff auf den Archivspeicher, sowie zum
- Abruf des Status laufender Importaufträge.

Die API soll die Integration in andere Systeme unterstützen. So ermöglicht die Import API die Integration des OLA-HD in bestehende OCR- und Digitalisierungsworkflows. Datenproduzenten müssen die Importpakete im OCRD-ZIP Format packen.

Ein im Rahmen des Projekts entwickeltes Packaging-Werkzeug[79] ermöglicht die einfache Erzeugung von Importpaketen und deren Übergabe an die Importschnittstelle des Archivs. Das Werkzeug analysiert die METS Strukturen, lädt die referenzierten Dokumente (z.B. Bilder in unterschiedlichen Auflösungen, Volltexte, etc.), speichert sie lokal gemäß OCRD-METS, korrigiert Referenzen im METS (FileSec), packt das Importpaket gemäß OCRD-ZIP und startet die Übergabe an die OLA-HD Importschnittstelle. Das Werkzeug kann als Demonstrator für die Integration dienen und in Digitalisierungsworkflows eingebunden werden.

[79] https://github.com/subugoe/OLA-HD-IMPL/tree/master/packaging-tool

Der Import eines Archivobjekts erfolgt über den Endpunkt /bag. Wenn der Import ohne Nennung einer Vorgänger ID erfolgt, wird ein neues Archivobjekt erzeugt. Bei Import mit Vorgänger ID mündet der Import in einer neuen Version eines bestehenden Archivobjekts.

Der Export aus dem Produktivspeicher kann direkt veranlasst werden (/export). Für den Export aus dem Archivspeicher muss das Archivobjekt zunächst für den Export angemeldet werden (/export-request), wobei das Archivspeicherobjekt im Hintergrund vom Band geladen und in einen Zwischenbereich auf der Festplatte abgelegt wird. In einem zweiten Schritt kann auf das Archivobjekt im Zwischenspeicher zugegriffen werden (/full-export).

7 Spezifikation zur technischen und wirtschaftlich-organisatorischen Umsetzung

Die Spezifikation zur technischen und wirtschaftlich-organisatorischen Umsetzung dient an erster Stelle als eine Entscheidungsgrundlage für die Produktisierung und die perspektivische Überführung des Dienstes in den Regelbetrieb sowie Aufnahme in das Service-Portfolio der Gesellschaft für Wissenschaftliche Datenverarbeitung (GWDG) und Niedersächsischen Staats- und Universitätsbibliothek (SUB).

7.1 Zielgruppendefinition und Stakeholder

Zu den Zielgruppen des Services gehören, wie im Abschnitt 3.1 Anwendergruppen behandelt;

- **Datenproduzenten**, z.B. Institutionen und wissenschaftliche Projekte mit Digitalisierungs- und Erschließungsaufgaben
- **Betreiber von Repositorien und Portalen,** z.B. Archive, Bibliotheken, Museen, Universitäre Einrichtungen, und andere Gedächtniseinrichtungen
- **Anwender**, z.B. wissenschaftliche Anwender und Projekte, die OCR-Daten verwenden möchten.

Zu den Stakeholdern gehören Förderer wie die DFG, das OCR-D Koordinierungsprojekt und die Modulprojekte, die Beiräte der VD-Projekte, die Entwickler der Digitalisierungsworkflowssoftware wie u.a. Goobi, Kitodo, Visual Library.

7.2 Erfolgskriterien

Die Beantwortung der Fragen zu den verschiedenen Aspekten der Erfolgskriterien[80] dient zur Identifizierung der möglichen Lücken bei dem Service-Konzept auf und zur Entscheidungsfindung, ob aus dem Prototyp ein Service entstehen kann.

- **Nutzung:** Welche Intensität und Umfang wird die Nutzung von OLA-HD durch die Zielgruppe haben? Unterscheiden sich bestimmte Komponenten hinsichtlich ihrer Nutzung? Wie viele Nutzer werden OLA-HD in Relation zur Gesamtgröße der

[80] vgl. http://resolver.sub.uni-goettingen.de/purl/?dariah-2014-5 ; s. 19

potenziellen Zielgruppe verwenden? Für welche Abschnitte ihrer Workflows verwenden die Nutzer OLA-HD?

- **Signifikanz:** Wie ist es um die Sichtbarkeit von OLA-HD innerhalb der potenziellen Nutzergruppe bestellt? Wie ist die Einschätzung hinsichtlich des Nutzens von OLA-HD für die Zielgruppe? Wird OLA-HD als nützlicher Beitrag gesehen? Hat OLA-HD Wirkung außerhalb der bekannten Zielgruppe?
- **Nachnutzung der Technik:** Werden Infrastrukturkomponenten von OLA-HD in anderen Zusammenhängen genutzt? Werden die Komponenten weiterentwickelt? Können sie bei Bedarf ausgetauscht werden?
- **Außenwirkung:** Wie erfolgreich ist die Öffentlichkeitsarbeit? Wie ist die Sichtbarkeit in der potenziellen Zielgruppe und Förderer? Verweisen die gängigen Richtlinien auf OLA-HD (z.B. DFG-Praxisregeln Digitalisierung)?
- **Interoperabilität:** Setzt OLA-HD die etablierten Standards und Datenformate ein? Sind die vorhandenen APIs für eine Maschine-zu-Maschine-Kommunikation mit den gängigen Software-Lösungen ausreichend?
- **Skalierbarkeit/Performance:** Kann OLA-HD mit wachsenden Nutzer- oder Objekt-Zahlen umgehen? Wie schnell können die Workflowschritte in OLA-HD durchgeführt werden?
- **Usability:** Erfüllt OLA-HD die Ansprüche der Nutzer an die Erlernbarkeit und Bedienbarkeit? Wie wird die Usability im Vergleich zu anderen Digitalisierungs- bzw. Archivierung-Werkzeugen bewertet?

7.3 Kostenmodell

Es gibt verschiedene Möglichkeiten für die Aufschlüsselung eines Kostenmodells.[81]

Üblicherweise beruhen Kostenmodelle auf der Berechnung der Kosten für jeden einzelnen Kunden.[82] Kunden können in diesem Zusammenhang sehr breit interpretiert werden. Kunden können interne Abteilungen sein, die Service von anderen internen Abteilungen entgegennehmen, aber auch externe Kunden, die Dienstleistungen von einer Organisation beauftragen. Insbesondere der unterschiedlich steigende Bedarf an Datenspeicher mit jedem neuen Kunden ist ein gutes Beispiel für Kosten dieser Art. Daher ist es sinnvoll diese Kosten nutzungsabhängig darzustellen.

Es kann aber auch sinnvoll sein die Berechnung der Kosten so zu gestalten, dass ein Kostenmodell Aussagen darüber treffen kann, wie viel eine spezielle Service-Bereitstellung

[81] vgl. DP4LiB Kostenmodell, Verfügbar unter: http://dp4lib.langzeitarchivierung.de/downloads/DP4lib-Kostenmodell_eines_LZA-Dienstes_v1.0.pdf . auch diskutiert in Schmitt, Karlheinz (2013): Kosten der digitalen Archivierung - Ein mögliches Vorgehensmodell und erste Erfahrungen. 16. Tagung des Arbeitskreises „Archivierung von Unterlagen aus digitalen Systemen": Digitale Archivierung in der Praxis, hg. v. Christian Keitel und Kai Naumann, 22-25. Stuttgart, Verlag W. Kohlhammer. Verfügbar unter: https://www.la-bw.de/media/full/69808

[82] DP4lip Kostenmodell zur Berechnung jährlicher Kosten des LZA-Services Verfügbar unter: http://dp4lib.langzeitarchivierung.de/downloads/DP4lib-LZA-Kosten-generisch.xlsx

kostet. In diesem Fall muss die Aufschlüsselung der Kosten so gestaltet werden, dass alle Kostenelemente den spezifischen Services zugeordnet werden können. Je nach Lebensphasen des IT-Services sind folgende direkte und indirekte Kostenfaktoren relevant.

- Personalkosten: Die Personalkosten stellen für die meisten IT-Services mit großem Abstand den größten Kostenfaktor dar, der bis zu 80% der Gesamtausgaben bilden kann. Dabei umfassen die direkten Personalkosten die Gehälter der Mitarbeiter, etwa für die Erst- und Fort-Entwicklung, den technischen Betrieb sowie den Nutzer-Support. Darüber hinaus entstehen Entwicklungskosten für wiederverwendete IT-Systeme (z.B. CDSTAR) und Verwaltungskosten für das Personal (indirekte Personalkosten).
- Betriebskosten außer Personal: Die Betriebskosten außer Personal umfassen die Bewirtschaftungskosten für die technische und organisatorische Infrastruktur und die Abschreibungen für große Investitionen. Hierunter fallen beispielsweise die Kosten für Strom, die Kälteversorgung bei größeren Rechneranlagen sowie Aufwendungen für die Gebäudenutzung (z.B. Instandhaltung und Miete). In diese Kategorie fällt ebenfalls die für den Betrieb erforderliche Softwarelizenzen und Hardware, etwa für die Netzwerkinfrastruktur (einschließlich Bandbreite) und den Datenspeicher.
- Sachkosten: Beispiele hierfür sind Mittel für die Durchführung von Weiterbildungen und Workshops, Dienstreisen, Verbrauchsmaterial, die Erstellung von Printmedien (z.B. in Form von Publikationen), Öffentlichkeitsarbeit sowie sonstige Fremdleistungskosten.

Kosten eines perspektivischen Services wie OLA-HD stellt sich wie folgt zusammen[83]:

Kosten OLA-HD =	Direktkosten +	Anteil an indirekter Kosten +	Nutzungsabhängige Kosten

7.4 Organisationsstrukturen

Für den Betrieb und Weiterentwicklung eines IT-Services mit Schichtenarchitektur[84] sind verschiedene Rollen und Kompetenzen erforderlich. Dazu zählen System-Management (inhaltlich / technisch), Support, Systemadministration für verschiedene Schichten der Architektur, Software-Engineering, Usability-Expertise. Diese sind zwischen Infrastrukturanbieter wie Rechenzentrum und Bibliothek unterschiedlich verteilt. Um dieses Problem zu begegnen und Redundanzen zu vermeiden, wurde 2014 die eResearch Alliance (eRA) als virtuelle Organisation ins Leben gerufen, die gemeinsam von Gesellschaft für Wissenschaftliche Datenverarbeitung (GWDG) und Niedersächsische Staats- und Universitätsbibliothek (SUB) als zentrale Anlaufstelle für Forscher, Forschungsverbände und Fakultäten betrieben wird. Universitätsmedizin Göttingen (UMG) ist seit 2015 der dritte Partner.

[83] vgl. Gadatsch / Mayer 2006: Masterkurs IT-Controlling, Andreas Gadatsch & Elmar Mayer, Vieweg Verlag, 3. Auflage, 2006; s.167-170

[84] Martin Fowler. 2002. Patterns of Enterprise Application Architecture. Addison-Wesley Longman Publishing Co., Inc., USA.

Bestehend aus den Service-Portfolios dieser Institutionen umfassen die Dienstleistungen von eRA Aspekte von eResearch über den gesamten Forschungslebenszyklus hinweg. Sie reichen von der Beratung zu Forschungsdatenmanagementplänen in der Vorbereitungsphase von Forschungsprojekten über die Anwendung digitaler Werkzeuge und Dienstleistungen in der eigentlichen Projektphase bis hin zu Publikationsmöglichkeiten, Archivierungslösungen und Abrufmöglichkeiten für Forschungsdaten am Ende von Projekten.

7.5 Technik

Wie bereits unter 6.1 erläutert, baut die Implementierung auf eine lose gekoppelte Architektur aus Einzelkomponenten auf, die je nach Nutzungsszenario ausgetauscht werden können. Dadurch kann das OLA-HD je nach Anwendungsfall und Voraussetzungen des Zielsystems flexibel eingesetzt werden.

Die mit der Importschnittstelle und dem Archiv-Manager realisierte Trennung von Produktiv- und Archivspeicher hilft Kosten, die bei einem Archiv naturgemäß dauerhaft entstehen, zu senken. Die vollständigen Daten werden in kostengünstigem Bandspeicher vorgehalten und sind dort, mit Einbußen bei der Zugriffszeit, verfügbar. Daten, auf die regelmäßig und schnell zugegriffen werden soll, werden im vergleichsweise kostenintensiven festplattenbasierten Produktivspeicher (Online-Speicher) gespeichert. Jeder Import erzeugt je ein Archiv- und ein Produktivspeicherobjekt. Die eingesetzte Infrastruktur nutzt ein **hierarchischen Speichermanagement (HSM)** [85], womit Daten (d.h. Dateien), auf die längere Zeit nicht zugegriffen wurde, bei Bedarf, d.h. um Platz zu schaffen, auf das Band ausgelagert werden. Auch hierdurch werden die Kosten gesenkt.

Kategorien	Schichten	Details
Infrastruktur-Services	Computer Service	VM (4 Cores, 8 GB RAM, 80 GB HDD)
	Storage Service	Produktivspeicher per NFS in die VM gemountet (3TB als prototypische Umgebung gewählt)
	Archive Service	Tape (empfohlen 20TB bei 3TB Produktivspeicher)
	ePIC PID Service	Für PIDs sind ePIC PIDs in den Prototypen integriert. Diese basieren auf dem Handle System und können bei Bedarf durch andere Handle-basierte Systeme ersetzt werden.
	Basisdienste	• Netzwerk/Datentransfer • IP Adresse, DNS • Nutzermanagement

[85] Eng. Hierarchical Storage Management (HSM)

Software/ Lizenzen	Host-Betriebssystem	Ubuntu-Lizenzfrei
	Datenbanken	CDSTAR, MongoDB, ElasticSearch - Alle lizenzfrei zu betreiben

Die prototypische Umsetzung wurde in einer Virtualisierungsumgebung (als Docker-Instanz) bereitgestellt, um weitere Evaluationen und Nachnutzungen zu ermöglichen. Die Softwarekomponenten wurden in Github[86] quelloffen zur Verfügung gestellt. Die entwickelte Software wurde unter der Apache Software License 2.0 und das schriftliche Konzept[87] sowie die Dokumentationen unter CC-BY-SA 4.0 zur Verfügung gestellt, um die freie Nachnutzbarkeit zu gewährleisten. Auf der OLA-HD Website[88] (Demo-Seite) kann das System getestet werden.

Der OLA-HD Service kann komplett in der eigenen Infrastruktur betrieben werden, die einzige externe Abhängigkeit ist der ePIC PID Service. Es gibt aber auch andere Anbieter in Europa, die diesen Dienst anbieten und auch eine Umstellung auf einen anderen Handle-basierten Dienst für PIDs ist möglich. So gesehen kann der OLA-HD Service auch, falls gewünscht, komplett ohne externe Abhängigkeiten betrieben werden.

8 Fazit

Im Rahmen des Projekts durchgeführten Expertengespräche und OCR-D Veranstaltungen bestand immer großes Interesse an OLA-HD Konzept und dessen prototypische Implementierung. Das bisherige Feedback hat verdeutlicht, dass der Bedarf für die Weiterentwicklung und Verbesserung von OLA-HD besteht, da die spezifische Archivierung von OCR-Ergebnissen ggf. gemeinsam mit Bildern und Metadaten noch nicht als Service vorhanden ist. Es wird weiteres Feedback im Rahmen der OCR-D Pilotierung erwartet.

SUB und GWDG erwägen den Prototyp weiterzuentwickeln und die Produktisierung im Rahmen eines Projekts voranzutreiben, so dass die Integration in ihr gemeinsames Portfolio und eine attraktive preisliche Gestaltung ermöglicht werden kann.

[86] https://github.com/subugoe/OLA-HD-IMPL

[87] https://github.com/subugoe/OLA-HD-IMPL/raw/master/docs/OLA-HD_Konzept.pdf

[88] http://ola-hd.gwdg.de, http://ola-hd.sub.uni-goettingen.de