SLUB
Wir führen Wissen.

SIP Spezifikation für automatischen Ingest SLUBArchiv

SLUB Dresden

Version 2.2.1, 2026-11-05

Inhaltsverzeichnis

Vorwort . . . 1
Überblick . . . 1
Warum der Wechsel auf BagIt? . . . 2
Aufbau SIP . . . 2
- Generelle Festlegungen . . . 2
- Bestandteile des SIP . . . 3

Komprimierte SIPs . . . 9
AIP Update . . . 9
- Metadaten Update . . . 9
- Volles Update . . . 9

Beispiele . . . 10
- SIP-Erstellung für Erstingest oder volles AIP-Update . . . 10
- SIP-Erstellung für reines Metadaten-Update . . . 12

Release Note 2.2.1 vom 2026-11-05

- Wegfall baginfo-Key "SLUBArchiv-rightsVersion"
- Anpassung der SIP Version

Vorwort

Dieses Dokument richtet sich an Produzenten, die digitale Objekte in das SLUBArchiv.digital einliefern und diese langfristig benutzbar erhalten wollen.

Überblick

Dieses Dokument beschreibt den Aufbau eines Ablieferungspaketes (englisch: Submission Information Package, SIP), welches für die automatische Verarbeitung zu archivierender digitaler Datenobjekte (IE) durch die Submission Application der SLUB und Einlieferung (Ingest) in das SLUBArchiv geeignet ist.

Dieses Dokument ist ein Teil der Übernahmespezifikation für das SLUBArchiv. Zur Übernahmespezifikation gehören die folgenden Dokumente [1]:

- In der **Übernahmevereinbarung** zwischen SLUB und Dienstnehmer sind die Daten, Ansprechpartner und organisatorischen Randbedingungen beschrieben. Dies schliesst die zu verwendenden Handreichungen für Dateiformate bzw. Objektgruppen mit ein.
- In der **Langzeitarchivfähige Dateiformate** sind die Formate aufgeführt, die die SLUB als potenziell archivfähig bewertet und für die die Funktionalitäten der Formaterkennung, Formatvalidierung und Metadatenextraktion in einem ausreichenden Maß durch das SLUBArchiv gewährleistet werden könnten. Die genaue Festlegung erfolgt spezifisch für jeden Workflow und jede Objektgruppe auf Basis der ermittelten signifikanten Eigenschaften.
- Die **SIP Spezifikation für automatischen Ingest SLUBArchiv** beschreibt den Aufbau der Ablieferungspakete (englisch: Submission Information Package, SIP) mit denen der Dienstnehmer die zu archivierenden Dokumente für das SLUBArchiv bereitstellt.
- Die **DIP Spezifikation für automatischen Access SLUBArchiv** beschreibt den Aufbau eines Auslieferungspaketes (englisch: Dissemination Information Package, DIP), welches für die automatische Weiterverarbeitung zielgruppengerechter Ausspielungen (Access) von im SLUBArchiv archivierter digitaler Datenobjekte (IE) geeignet ist
- Die **Workflow Spezifikation für automatisierte Interaktionen mit dem SLUBArchiv** beschreibt den Prozess der Übergabe zu archivierender Dokumente in das SLUBArchiv (Ingest / AIP Update), das Fehlerprotokoll und den Zugriff auf die archivierten digitaler Objekte (Access).
- Das Dokument **Spezifikation Rechteauszeichnung SLUBArchiv** beschreibt, wie rechtliche Informationen zu einem Datenobjekt kodiert und abgelegt werden müssen.
- Das Dokument **Webservice SLUBArchiv** beschreibt Funktionen, die Dienstnehmer nutzen können, um Informationen über ihre Daten im SLUBArchiv über einen Webservice abzufragen.
- Vom SLUBArchiv verwendete Begriffe sind im **Glossar SLUBArchiv** definiert.

Warum der Wechsel auf BagIt?

Das SLUBArchiv hat sehr gründlich die Vor- und Nachteile des Wechsels des Formates der Submission Information Packages (SIP) von einer eigenen, auf METS basierenden, hin zu einer auf BagIt [https://tools.ietf.org/html/rfc8493] basierenden Variante abgewogen.

Für den Wechsel sprechen mehrere Gründe:

- BagIt [https://tools.ietf.org/html/rfc8493] ist mittlerweile in Version 1.0 als RFC [https://www.rfc-editor.org] standardisiert.
- Immer mehr Archivlösungen setzen auf BagIt als Einlieferungsformat (z. B. Archivematica und Rosetta).
- Es gibt bereits eine große Anzahl an Programmen und Programmbibliotheken, die Funktionalitäten zur einfachen Verarbeitung von BagIt bereitstellen, hier eine kleine Auswahl:
  - libcbag [http://andreas-romeyke.de/software.html#_libcbag_a_free_and_opensource_c_library_to_handle_bagit_structures] C++-Bibliothek
  - pybagit [https://pypi.python.org/pypi/pybagit/] Python-Modul
  - Archive::BagIt [http://metacpan.org/module/Archive::BagIt] Perl-Modul
  - Bagger [https://github.com/LibraryOfCongress/bagger] GUI-Programm
  - bagit-python [https://github.com/LibraryOfCongress/bagit-python] CLI-Programm
  - bagit-java [https://github.com/LibraryOfCongress/bagit-java] Java-Bibliothek
- Das Datenformat ist einfach aufgebaut und kann mit Bordmitteln erzeugt und verarbeitet werden.
- Durch die Standardisierung von BagIt [https://tools.ietf.org/html/rfc8493] und weite Verbreitung erfolgt eine Normierung der Werkzeuge für die Erstellung, Prüfung und Verarbeitung von SIPs.
- Die Hürden für die Arbeit mit BagIt sind geringer als die von METS-basierten SIPs.
- BagIt [https://tools.ietf.org/html/rfc8493] ist modular aufgebaut und erlaubt die einfache Extraktion von Bestandteilen des SIPs.

Aufbau SIP

Generelle Festlegungen

Jedes SIP besteht aus einem Verzeichnis mit einem eindeutigen Namen (z. B. Zeitstempel der Erzeugung), welches die BagIt [https://tools.ietf.org/html/rfc8493]-Verzeichnisstruktur abbildet. Der Verzeichnisname wird nicht ausgewertet, sollte aber eindeutig sein, damit es nicht zu Überschreibungen durch die Abliefernden kommt.

Der Produzent ist in der Verantwortung, ein SIP mit korrekter BagIt [https://tools.ietf.org/html/rfc8493]-Implementierung zu erstellen.

Jedes SIP enthält genau eine Intellektuelle Einheit (IE). Dies vereinfacht die Prozessierung im

SLUBArchiv.

Eine Intellektuelle Einheit ist ein Datenobjekt, welches aus einer oder mehreren Dateien besteht, die gemeinsam verwaltet und charakterisiert werden.

Alle Dateien dieser IE müssen innerhalb der BagIt [https://tools.ietf.org/html/rfc8493]-Struktur referenziert sein.

Das SLUBArchiv unterscheidet zwei Arten von SIPs. Ein SIP kann

- entweder die Daten für die erstmalige Aufnahme einer IE (Erstingest),
- oder die Daten für eine Aktualisierung (Update) der IE (AIP Update)

enthalten.

Bestandteile des SIP

Grundlage

Basis des SIPs ist eine BagIt [https://tools.ietf.org/html/rfc8493]-Struktur nach RFC8493. Ergänzend gelten folgende Festlegungen.

Deskriptive Metadaten

Metadaten des Bags werden als Schlüssel-Werte-Paare in der 'bag-info.txt'-Datei kodiert (sh. 2.2.2, RFC 8493).

Es sollten nur diejenigen deskriptiven Metadaten übernommen werden, die minimal notwendig sind, um einen rudimentären Katalog aufbauen zu können.

Es wird empfohlen, die Angaben zur dienstnehmenden Institution und zu persistenten Identifiern in den beschreibenden Metadaten zu kodieren. Dazu sind die vorgesehenen Felder nach BagIt [https://tools.ietf.org/html/rfc8493]-Spezifikation zu nutzen (z. B. 'Source-Organization', 'External-Description', ...).

Ergänzend sollten, wenn möglich, noch der Titel des Dokumentes ('Title') und die Verfasser ('Author') als Schlüssel-Werte-Paare kodiert werden.

Beispiel MODS als Ausgangspunkt

```
<mods:mods xmlns:mods="http://www.loc.gov/mods/v3">
    <mods:location>
        <mods:physicalLocation authority="marcorg" displayLabel="Saxon State Library,
Dresden, Germany">DE-14</mods:physicalLocation>
        <mods:shelfLocator>Hist.Sax.M.37.t,120</mods:shelfLocator>
    </mods:location>
    <mods:relatedItem type="series">
        <mods:titleInfo>
            <mods:title>Saxonica</mods:title>
        </mods:titleInfo>
```

```
    </mods:relatedItem>
    <mods:recordInfo>
        <mods:recordIdentifier source="http://digital.slub-dresden.de/oai/">oai:de:slub-
dresden:db:id-319037843</mods:recordIdentifier>
    </mods:recordInfo>
    <mods:physicalDescription>
        <mods:digitalOrigin>reformatted digital</mods:digitalOrigin>
        <mods:extent>[1] Bl.</mods:extent>
    </mods:physicalDescription>
    <mods:identifier type="urn">urn:nbn:de:bsz:14-db-id3190378431</mods:identifier>
    <mods:titleInfo>
        <mods:title>Eingabe der Handelskammer zu Leipzig den Entwurf eines Tabak-Steuer-
Gesetzes betr.</mods:title>
        <mods:subTitle>an den Reichstag zu Berlin</mods:subTitle>
    </mods:titleInfo>
    <mods:language>
        <mods:languageTerm authority="rfc3066" type="code">de</mods:languageTerm>
    </mods:language>
    <mods:originInfo>
        <mods:place>
            <mods:placeTerm type="text">[Leipzig]</mods:placeTerm>
        </mods:place>
        <mods:dateIssued keyDate="yes">1893</mods:dateIssued>
    </mods:originInfo>
    <mods:subject authority="slub">
        <mods:topic>eingdehaz</mods:topic>
    </mods:subject>
</mods:mods>
```

Beispiel Mapping des MODS auf 'bag-info.txt'

```
External-Identifier: oai:de:slub-dresden:db:id-319037843
External-Identifier: urn:nbn:de:bsz:14-db-id3190378431
Title: Eingabe der Handelskammer zu Leipzig den Entwurf eines Tabak-Steuer-Gesetzes betr.
an den Reichstag zu Berlin
…
```

Es wird empfohlen, Metadaten, die zu einer zu archivierenden Intellektuellen Einheit gehören, neben der Kodierung in den Metadaten des BagIt **zusätzlich** im originalen Metadatenformat als Datei im Unterordner 'meta/' abzulegen, siehe Abschnitt Ablage Metadaten-Dateien .

Die beschreibenden Metadaten, die originärer Bestandteil des BagIt [https://tools.ietf.org/html/rfc8493] sind (sh. 'bag-info.txt'), werden u.a. für die Suche innerhalb des Archivs verwendet und müssen mit den eventuell beigelegten Metadaten-Dateien (unter 'meta/') inhaltlich übereinstimmen.

Das SLUBArchiv sichert die Langzeitverfügbarkeit des IEs einer SIP via Formatmigration.

Für die Metadaten unter 'meta/' wird **keine** Formatmigration durchgeführt!

Administrative Metadaten für die Steuerung des Ingests

Ebenfalls in der Datei 'bag-info.txt' müssen **verpflichtend** nachfolgende Informationen als Schlüssel-Werte-Paare kodiert werden. Wiederholungen sind nicht zulässig.

Der Produzent ist die Rolle eines Dienstnehmers, der eine IE archivieren möchte und ein SIP an das SLUBArchiv einliefert. Die nachfolgenden Daten beziehen sich auf den Workflow des Produzenten, unabhängig davon, ob er im Auftrag Dritter handelt.

SLUBArchiv-sipVersion

Der Schlüssel 'SLUBArchiv-sipVersion' enthält 'v2026.2', für die eindeutige Identifizierung des SIP-Formates des SLUBArchivs.

SLUBArchiv-externalWorkflow

In diesem Schlüssel wird der Name des Übergabeworkflows des Dienstnehmers angegeben, aus dem die zu archivierenden Daten stammen. Zusammen mit der 'SLUBArchiv-externalId' wird diese Information im SLUBArchiv genutzt, um Archivpakete eindeutig zu identifizieren. Anhand dieser beiden Einträge können SIPs zu einem digitalen Dokument (einer IE) zugeordnet werden. Dies ermöglicht, dass Archivpakete aktualisiert werden können.

Der Name muss eindeutig sein, sowie sich aus den Zeichen 'a-z0-9_-' zusammensetzen. Großschreibung wird nicht unterstützt. Empfohlen wird einen sprechenden Namen zu verwenden.

SLUBArchiv-externalId

In diesem Schlüssel muss eine eindeutige ID innerhalb eines Workflows durch den Produzenten vergeben werden. Diese ID und der Workflowname im Tag 'SLUBArchiv-externalWorkflow' identifizieren Archivpakete im SLUBArchiv eindeutig (siehe Beschreibung zum Tag 'SLUBArchiv-externalWorkflow'). Der Identifier muss persistent sein. Eine automatische Ersetzung eines bestehenden externen Identifiers über die Submission Application ist nicht vorgesehen.

Der externe Identifier darf sich nur aus den Zeichen 'a-z0-9_-' zusammensetzen. Großschreibung wird nicht unterstützt.

Dieses Feld ist nicht unbedingt identisch mit dem in der BagIt [https://tools.ietf.org/html/rfc8493]-Spezifikation angegebenen Schlüssel 'External-Identifier'. Es empfiehlt sich, beide identisch zu belegen.

SLUBArchiv-externalIsilId

Dieser Schlüssel sollte von Bibliotheken und andere Gedächtnisorganisationen mit ISIL Nummer genutzt werden. Er erlaubt die Zuordnung von Vorgängen zu Einrichtungen. Für weitere Informationen nutzen Sie bitte https://sigel.staatsbibliothek-berlin.de/. Einrichtungen ohne ISIL-Nummer verwenden das Tag nicht.

Der Schlüssel bezieht sich auf die Organisationseinheit des Produzenten, unabhängig davon, ob er im Auftrag Dritter handelt.

SLUBArchiv-exportToArchiveDate

Dieses Tag gibt in ISO-Notation (ISO 8601) das Datum an, an dem das konkrete SIP vom Abliefernden für das SLUBArchiv erstellt wurde. Da ein SIP sowohl für eine erstmalige Übernahme ins Langzeitarchiv als auch für ein Update verwendet werden kann, muss die richtige Reihenfolge der Übernahme von SIPs zu einem Dokument sichergestellt werden. Das Datum in diesem Tag (und nur dieses Datum) regelt, in welcher Reihenfolge mehrere SIPs, die zur gleichen Intellektuellen Einheit (IE) gehören, abgearbeitet werden. Wird das Datum nicht korrekt gesetzt, können durch eine falsche Abarbeitungsreihenfolge der Änderungen falsche Archival Information Packages (AIPs) entstehen. Es ist daher erforderlich, das Datum sekundengenau zu setzen.

Dieses Feld ist nicht unbedingt identisch mit dem in der BagIt [https://tools.ietf.org/html/rfc8493]-Spezifikation angegebenen Schlüssel 'Bagging-Date'. Der Schlüssel 'SLUBArchiv-exportToArchiveDate' erlaubt eine sekundengenaue Angabe, die in 'Bagging-Date' nicht vorgesehen ist. Es wird empfohlen 'Bagging-Date' mit den identischen Jahr-Monat-Tag-Angaben aus dem 'SLUBArchiv-exportToArchiveDate' zu belegen.

SLUBArchiv-hasConservationReason

Dieses Feld wird mit 'true' belegt, wenn die zu archivierende Intellektuelle Einheit aus Bestandserhaltungsgründen langzeitarchiviert wird. Dies meint, dass es zu dieser IE keine physische oder anderweitige Vorlage (mehr) gibt.

In allen anderen Fällen ist das Feld mit 'false' zu belegen.

SLUBArchiv-archivalValueDescription

Dieses Feld beschreibt, **warum** die Intellektuelle Einheit (IE) langzeitarchiviert werden soll. Diese Angabe hilft zu entscheiden, wie bei einer Formatmigration vorgegangen werden soll. Sie enthält eine intellektuelle Bewertung der Daten hinsichtlich ihrer Archivwürdigkeit. Gründe für die Archivwürdigkeit können sich aus gesetzlichen Vorgaben oder aus besonderen kulturellen oder wissenschaftlichen Werten ergeben.

Dieses Feld ist nicht mit dem in der BagIt [https://tools.ietf.org/html/rfc8493]-Spezifikation angegebenen Schlüssel 'External-Description' zu verwechseln, welcher den Inhalt des Bags und dessen Herkunft beschreibt.

Beispiel

'bag-info.txt' mit notwendigen Steuerungsdaten

```
SLUBArchiv-sipVersion: v2026.2
SLUBArchiv-exportToArchiveDate: 20160101T120000.00
SLUBArchiv-externalId: 10008
SLUBArchiv-externalIsilId: DE-14
SLUBArchiv-externalWorkflow: kitodo
SLUBArchiv-hasConservationReason: true
SLUBArchiv-archivalValueDescription: Gesetzlicher Auftrag der SLUB Dresden
…
```

Rechteinformationen

Die genaue Beschreibung ist im Dokument **Spezifikation Rechteauszeichnung SLUBArchiv** zu finden.

Die XML-Datei mit den SLUBArchiv-Rechteinformationen ist als Tagfile 'rights.xml' unter dem Verzeichnis 'meta/' abzuspeichern (sh. 2.3 BagIt [https://tools.ietf.org/html/rfc8493]-Spezifikation).

Die Datei muss in den BagIt [https://tools.ietf.org/html/rfc8493]-Dateien 'tagmanifest-XXXX.txt' hinterlegt sein, wobei XXXX den verwendeten Prüfsummenalgorithmen entspricht (sh. 2.4 BagIt [https://tools.ietf.org/html/rfc8493]-Spezifikation).

Prüfsummen

Als Prüfsummen sind gemäß BagIt [https://tools.ietf.org/html/rfc8493]-Spezifikation (2.4.) mindestens die beiden folgenden zu verwenden:

- sha-512
- md5

Ebenso ist das Tag 'Payload-Oxum' in der 'bag-info.txt' zu belegen (sh. 2.2.2, BagIt [https://tools.ietf.org/html/rfc8493]-Spezifikation).

Die Berechnung der 'Payload-Oxum' ist nur für die Dateien im 'data/'-Verzeichnis anzuwenden.

Zu archivierende Dateien und zugehörige Prüfsummen

Alle Dateien, die zur eigentlichen IE gehören, sind unterhalb des 'data/'-Verzeichnisses abzuspeichern (sh. BagIt [https://tools.ietf.org/html/rfc8493], Abschnitte "Payload" 2.1.2, 2.1.3)

Ablage Metadaten-Dateien

Es können Metadaten-Dateien, die Zusatzinfos über das IE enthalten, in dem Verzeichnis 'meta/' abgelegt werden.

Metadaten, die nicht die IE selbst, sondern Teile innerhalb einer IE beschreiben, sind nicht Bestandteil des 'bag-info.txt' oder der Metadaten-Dateien unter 'meta/', sondern müssen zur IE selbst hinzugerechnet werden. In diesem Fall ist zu beachten, dass

- diese Metadaten das Kriterium der Archivwürdigkeit erfüllen (Stichwort: Signifikante Eigenschaften)
- diese Metadaten-Dateien ebenfalls in einem langzeitarchivfähigem Dateiformat beschrieben sind
- im Fall einer Formatmigration durch das SLUBArchiv diese Metadatendateien einer Anpassung bedürfen und diese geregelt sein muss (Stichwort: Führendes System)

> Dies gilt z. B. für Strukturierungs- und Paginierungsinformationen.

Dazu können auch mehrere Metadatendateien genutzt werden. Zulässig sind dabei nur standardisierte, öffentlich zugängliche und validierbare Metadatenformate.

Diese Dateien werden vom Archiv validiert, darüber hinaus **nicht** ausgewertet (maßgebend bleibt die BagIt [https://tools.ietf.org/html/rfc8493]-Struktur). Sie gehören aber zu einem vollständigen Ablieferungspaket und erlauben den Erhalt der logischen Verknüpfung zwischen den zu archivierenden Nutzdaten und den zugehörigen Metainformationen außerhalb des SLUBArchiv.

Sie sind analog zu den SLUBArchiv-Rechteinformationen (sh. Rechteinformationen) in den Tag-Manifest-Dateien zu referenzieren.

> Diese XML-Dateien sollten möglichst einfach nach ihrem verwendeten Metadatenschema benannt werden. Für MODS-Dateien empfiehlt sich 'mods.xml', für LIDO 'lido.xml' und für Dublin Core 'dc.xml'. Diese Dateinamen werden **nicht** archiviert. Die Inhalte dieser Metadaten-Dateien werden im SLUB-Archiv in speziellen Datenfeldern abgelegt und können bei der Konstruktion von Abgabepaketen (DIP) abgefragt werden.

> Es steht dem Produzenten frei, in der 'bag-info.txt' die Informationen zu hinterlegen, die eine Interpretation dieser Metadaten erleichtern.

> Der Dateiname 'rights.xml' ist reserviert für die Rechteauszeichnung nach der SLUB-Spezifikation "Rechtekodierung".

> Das Verzeichnis 'meta/' ist optional. Wird es verwendet, so **müssen** seine Dateien in den 'tagmanifest-XXX.txt'-Dateien mit den korrekten Prüfsummen hinterlegt sein.

Weitere Hinweise

Die BagIt-Spezifikation erlaubt an einigen Stellen optionale Elemente. Im Folgenden sind diejenigen aufgelistet, die für die Abgabe an das SLUBArchiv verpflichtend sind:

- Keine Verwendung von unvollständigen BagIts mit Fetch-Option (sh. 2.2.3, BagIt [https://tools.ietf.org/html/rfc8493]-Spezifikation)
- 'Tag-File-Character-Encoding' in der 'bagit.txt'-Datei ist immer "UTF-8"
- Alle Tag-Dateien sind immer als UTF-8 (ohne BOM) zu kodieren.
- Dateipfade dürfen **keine** Leerzeichen enthalten.
- Die Tag-Manifest-Dateien sind verpflichtend und müssen jeweils die gleichen Tag-Dateien beschreiben.
- Die folgenden Schlüssel-Werte-Paare der Bag-Metadata-Datei 'bag-info.txt' sind zu belegen:
  - 'Bag-Size'
  - 'Payload-Oxum'
- Da ein SIP genau ein IE enthält, dürfen die folgenden Schlüssel-Werte-Paare der Bag-Metadata-Datei 'bag-info.txt' nicht benutzt werden:

- 'Bag-Count'
- 'Bag-Group-Identifier'

Komprimierte SIPs

Komprimierte SIPs werden **nicht** unterstützt. Hintergrund ist, dass die Verarbeitung komprimierter SIPs aufwendiger und fehleranfälliger ist.

AIP Update

Das SIP für ein AIP Update unterscheidet sich nicht wesentlich von einem SIP für einen Erstingest. Ein SIP für ein AIP Update ist ein SIP, dessen 'SLUBArchiv-externalWorkflow' und 'SLUBArchiv-externalId' auf ein bereits im SLUBArchiv vorhandenes IE verweist.

Im Falle eines AIP Update sind immer **alle** Metadaten im SIP zu kodieren!

Die ursprünglichen Metadaten werden im AIP vollständig ersetzt.

Die Besonderheiten werden nachfolgend erläutert.

Metadaten Update

Das Metadaten-Update umfasst ausschließlich Änderungen der in der 'bag-info.txt' kodierten Information oder Änderungen der in 'meta/' liegenden Metadaten-Dateien, die das im Archiv hinterlegte AIP aktualisieren.

Diese SIPs dürfen daher keine Dateien im 'data/'-Verzeichnis enthalten.

Auch wenn das Bag keine Dateien im 'data/' enthalten darf, **muss** das Verzeichnis 'data/' existieren. Gleiches gilt für die 'manifest'-Dateien, die dann leer sind.

Das Verzeichnis 'meta/' ist optional. Wird es verwendet, **müssen** seine Dateien in den 'tagmanifest-XXX.txt'-Dateien mit den korrekten Prüfsummen hinterlegt sein.

Um sicherzustellen, dass keine Inkonsistenzen auftreten können, dürfen die in 'meta/' hinterlegten Metadaten-Dateien **keine** Referenzen auf Dateien des Archivs bzw. auf Dateien, die in vorherigen Versionen des SIPs übermittelt wurden, enthalten. Andernfalls ist ein Volles Update durchzuführen.

Für ein tieferes Verständnis ist die Handreichung der SLUB "Workflow-Spezifikation" zurate zu ziehen, die die verschiedenen Optionen und deren Konsequenzen behandelt.

Volles Update

Das SIP enthält alle zum IE gehörenden Dateien in der neuen Fassung. Das SLUBArchiv ordnet die im SIP referenzierten Dateien dem vorhanden AIP zu und erzeugt über den AIP-Update-Prozess eine neue Version des AIP.

Das SLUB-Archiv erkennt über die in 'bag-info.txt' angegebenen Schlüssel 'SLUBArchiv-externalWorkflow' und 'SLUBArchiv-externalId', welches AIP im Archiv aktualisiert werden muss. Die Reihenfolge der Updates ist durch den Schlüssel 'SLUBArchiv-exportToArchiveDate' sichergestellt.

Beispiele

SIP-Erstellung für Erstingest oder volles AIP-Update

Als Eingangsbeispiel existiere folgende IE:

Verzeichnisstruktur IE

```
exampleIE/
├── 1.txt
├── 3.dat
└── subdir/
    ├── 2.png
    └── 2.mdx
```

Zu dieser IE existiere eine METS-MODS-Datei 'IE-METS-MODS.xml'.

Um ein SIP zu erstellen, sind folgende Schritte notwendig:

1. Ein Verzeichnis mit eindeutigem Namen wird erzeugt, welches das SIP repräsentiert.
2. Ein Unterverzeichniss 'data/' wird für die eigentliche Payload erzeugt. Die Inhalte des IE werden aus 'exampleIE/' in dieses Verzeichnis kopiert.
3. Das Verzeichnis 'meta/' wird erzeugt.
4. In 'meta/' wird eine gültige XML-Datei 'mods.xml' geschrieben, die (z. B. mittels XSLT) die reinen MODS-Daten der obigen 'IE-METS-MODS.xml' enthält.
5. Unter 'meta/' wird die Datei 'rights.xml' erzeugt, die Rechteinformationen für das IE gemäß SLUBArchiv **Spezifikation Rechteauszeichnung SLUBArchiv** enthält.
6. Die Datei 'bagit.txt' wird angelegt.
7. Die Dateien 'manifest-sha512.txt' und 'manifest-md5.txt' werden erzeugt. Diese enthalten ausschließlich Daten über die Dateien unter 'data/'.
8. Die Datei 'bag-info.txt' wird befüllt. Ebenso wird das Schlüssel-Werte-Paar 'Payload-Oxum' korrekt belegt.
9. Die Tag-Manifest-Dateien 'tagmanifest-sha512.txt' und 'tagmanifest-md5.txt' werden erzeugt. Diese enthalten neben den Tag-Files die Dateien unter 'meta/'.

Verzeichnisstruktur Bag

```
examplebag/
├── bag-info.txt
├── bagit.txt
```

```
├── data/
│   ├── 1.txt
│   ├── 3.dat
│   └── subdir/
│       ├── 2.png
│       └── 2.mdx
├── manifest-md5.txt
├── manifest-sha512.txt
├── meta/
│   ├── mods.xml
│   └── rights.xml
├── tagmanifest-md5.txt
└── tagmanifest-sha512.txt
```

Im Folgenden sind beispielhaft **einige** der BagIt-Dateien dieses Bags aufgelistet.

Inhalt der 'bagit.txt'

```
BagIt-Version: 1.00
Tag-File-Character-Encoding: UTF-8
```

Inhalt der 'bag-info.txt'

```
Author: Max Mustermann
Bagging-Date: 2015-12-28
Bag-Size: 389 kB
External-Description: Dies ist ein kleines Beispiel für
eine IE, die als SIP im BagIt-Format des SLUBArchivs eingeliefert
werden soll.
External-Identifier: testbag-01
Payload-Oxum: 388743.4
SLUBArchiv-archivalValueDescription: Gesetzlicher Auftrag der SLUB Dresden
SLUBArchiv-exportToArchiveDate: 20160101T120000.00
SLUBArchiv-externalId: 99193991991991
SLUBArchiv-externalIsilId: DE-14
SLUBArchiv-externalWorkflow: kitodo
SLUBArchiv-hasConservationReason: true
SLUBArchiv-sipVersion: v2026.2
Title: BeispielIE
```

Inhalt der 'manifest-sha512.txt'

```
cf83e1357eefb8bdf1542850d66d8007d620e4050b5715dc83f4a921d36ce9ce47d0d13c5d85f2b0ff8318d28
77eec2f63b931bd47417a81a538327af927da3e  data/subdir/2.png
a76c1de5fab3e09bc8f50225018da99a17c91addc1851ff7b201d41541a36cabe2c8b35f0477511368ab97e2d
a7fcb7a079a17379efb2a29c410454bb2fd9083  data/subdir/2.mdx
052cf2a5a608ce906d08d0d59d85d33b4d324cf0f14822aef727e700edd9dccfe6eb3613e0e32f047e5f36cfd
0a67634325253d6c626eb6d3f3f74b28fe3903d  data/1.txt
cf83e1357eefb8bdf1542850d66d8007d620e4050b5715dc83f4a921d36ce9ce47d0d13c5d85f2b0ff8318d28
77eec2f63b931bd47417a81a538327af927da3e  data/3.dat
```

Inhalt der 'manifest-md5.txt'

```
d41d8cd98f00b204e9800998ecf8427e  data/subdir/2.png
227bc609651f929e367c3b2b79e09d5b  data/subdir/2.mdx
e1cbb0c3879af8347246f12c559a86b5  data/1.txt
d41d8cd98f00b204e9800998ecf8427e  data/3.dat
```

Inhalt der 'tagmanifest-md5.txt'

```
d879078aefbe30394bf4fbf602daaae2  manifest-sha512.txt
7cb81a65369e7bbb9794971f2f1baeff  manifest-md5.txt
9ca92ea8aabf7c50b61d24422748f1ad  bag-info.txt
eaa2c609ff6371712f623f5531945b44  bagit.txt
63d20275d78e2dfddc135d8c6acbae15  meta/rights.xml
d41d8cd98f00b204e9800998ecf8427e  meta/mods.xml
```

SIP-Erstellung für reines Metadaten-Update

Es sei angenommen, dass das SIP für den Erstingest wie folgt aussah:

Verzeichnisstruktur Bag Erstingest

```
examplebag/
├── bag-info.txt
├── bagit.txt
├── data/
│   ├── 1.txt
│   ├── 3.dat
│   └── subdir/
│       ├── 2.png
│       └── 2.mdx
├── manifest-md5.txt
├── manifest-sha512.txt
├── meta/
│   ├── mods.xml
│   └── rights.xml
├── tagmanifest-md5.txt
└── tagmanifest-sha512.txt
```

Nun soll der Titel der IE von "BeispielIE" auf "BeispielIE2" geändert werden. Der Titel ist einmal als Schlüssel-Werte-Paar unter 'Title' in der Datei 'bag-info.txt' als auch in der Datei 'mods.xml' kodiert. Da die Datei 'mods.xml' keine Dateireferenzen enthält, wird für das Metadaten-Update folgendes SIP an das SLUB-Archiv übergeben:

Verzeichnisstruktur Bag Metadaten-Update

```
examplebag/
├── bag-info.txt
├── bagit.txt
├── data/
```

```
├── manifest-md5.txt
├── manifest-sha512.txt
├── meta/
│   ├── mods.xml
│   └── rights.xml
├── tagmanifest-md5.txt
└── tagmanifest-sha512.txt
```

Die Submission-Application stellt in dem Fall sicher, dass die neue Datei 'mods.xml' im AIP abgebildet wird und der Titel in den descriptiven Metadaten des AIP, (die das SLUBArchiv intern nutzt) auf "BeispielIE2" abgeändert wird.

Die eigentliche IE mit all ihren Dateien wird im AIP nicht geändert:

Verzeichnisstruktur IE nach Metadaten Update

```
exampleIE/
├── 1.txt
├── 3.dat
└── subdir/
    ├── 2.png
    └── 2.mdx
```

Der Inhalt der geänderten 'mods.xml' wird in spezifischen SLUBArchiv-Datenfeldern zum jeweiligen AIP gespeichert und kann bei der DIP-Konstruktion abgefragt werden.

[1] Die genannten Dokumente sind auf der Webseite des SLUBArchivs unter https://slubarchiv.slub-dresden.de/technische-standards-fuer-die-ablieferung-von-digitalen-dokumenten/ veröffentlicht und sind dort, technisch bedingt, spezifischer benannt.