



Anwendungshandbuch

Inhalt

1	Vorwort	2
2	Systemvoraussetzungen	3
3	Offene Punkte / Feedback	3
4	Einleitung duale PDF/A-Validierung	4
5	Installation von KOST-Val	5
6	Konfiguration von KOST-Val	5
6.1	Bestandteile der Konfigurationsdatei «kostval.conf.xml»	5
6.2	«KaD_SignatureFile_V72.xml» und «jhove.conf»	7
7	Ressourcen von KOST-Val	7
8	Validierung starten	8
8.1	KOST-Val GUI	8
8.2	Validierung manuell starten	10
9	Beschreibung der Validierungsschritte	12
9.1	TIFF	12
9.2	SIARD	13
9.3	PDF/A	14
9.4	JP2	15
9.5	JPEG	15
9.6	SIP	16
10	Urheberrecht	17
10.1	3-Heights™ PDF/A Validator API-Lizenz	18
10.2	pdfaPilot CLI Lizenz	19
11	Anhang	20
11.1	Programmaufbau	20
11.2	Funktionsprinzip Formatvalidierung	22

1 Vorwort

KOST-Val ist eine java-basierte Anwendung zur Validierung von Aufbau und Inhalt von TIFF-Dateien (*Tagged Image File Format*), SIARD-Dateien (*Software Independent Archiving of Relational Databases*), PDF/A-Dateien, JP2-Dateien (JPEG 2000; *Joint Photographic Experts Group 2000*) JPEG-Dateien sowie von sogenannter Submission Information Package (SIP) zur Ablieferung von digitalen Informationen. Diese Anwendung steht unter der GPL3+ Lizenz und wird durch die KOST der Öffentlichkeit quelloffen zur Verfügung gestellt. KOST-Val stützt sich auf unveränderte Komponenten anderer Hersteller, welche direkt im Quellcode von KOST-Val eingebunden sind. Die Benutzer von KOST-Val sind gehalten, die Lizenzbestimmungen all dieser Komponenten zu befolgen. Ausführliche Informationen sind im Kapitel 10 ersichtlich.

KOST-Val erfüllt die im Folgenden beschriebenen Anforderungen.

TIFF-Validierung: KOST-Val liest ein TIFF und validiert mit Hilfe von Jhove den Aufbau und Inhalt sowie mit ExifTool die zentralen Eigenschaften wie z.B. Komprimierung, Farbraum und Multipage. Die Eigenschaften sind konfigurierbar.

SIARD-Validierung: KOST-Val liest ein SIARD (eCH-0165¹ v1 und SIARD v2.1²) und validiert den Aufbau und Inhalt wie z.B. Struktur, Header- und Content-Validierung.

PDF/A-Validierung: KOST-Val liest ein PDF respektive PDF/A (ISO 19005-1 und 19005-2) und validiert mit Hilfe des 3-Heights™ PDF/A Validator von PDF-Tools oder auch des pdfaPilot von callas den Aufbau und Inhalt der PDF-Datei. KOST-Val gliedert die verschiedenen Fehlermeldungen in Hauptgruppen wie z.B. Schriften, Grafiken und Metadaten. Im Lieferumfang von KOST-Val sind einzig eingeschränkte³ Versionen von 3-Heights™ PDF/A Validator von PDF-Tools sowie pdfaPilot von callas enthalten.

Im Modul J kann konfiguriert werden, ob die JBIG2-Komprimierung⁴ akzeptiert wird oder nicht. Im Modul K wird je nach Konfiguration überprüft ob die Schriften durchsuch- und extrahierbar sind oder nicht.

JP2-Validierung: KOST-Val liest ein JP2 (ISO 15444) und validiert mit Hilfe von Jpylyzer den Aufbau und Inhalt.

JPEG-Validierung: KOST-Val liest ein JPEG (ISO 10918-1) und validiert mit Hilfe von BadPeggy den Aufbau und Inhalt⁵.

¹ Die Spezifikation kann von der eCH Website heruntergeladen werden:
<https://www.ech.ch/echech-0165>.

² Die Spezifikation kann von der KOST Website heruntergeladen werden:
https://kost-ceco.ch/cms/index.php?siard_de.

³ Die Einschränkung bezieht sich hauptsächlich auf den Durchsatz von bis zu 72'000 Seiten pro Jahr. Mehr Informationen sind in den Lizenzen in Kapitel 10 ersichtlich. Diese Einschränkung kann aufgehoben werden, in dem eine 3-Heights™ PDF/A Validator API Lizenz bei PDF-Tools erworben und mit dem LicenseManager (resources\3-Heights_PDF-Validator_dll) aktiviert respektive pdfaPilot bei callas erworben wird.

⁴ Zur Problematik der JBIG2-Komprimierung siehe http://kost-ceco.ch/cms/index.php?jbig2-compression_de. Die Preservation-Planning-Expertengruppe PPEG der KOST empfiehlt, beim Erstellen von PDF-Dateien vorerst auf die Kompressionsart JBIG2 zu verzichten.

⁵ Die Fehlermeldung „Not a JPEG file“ wird jedoch von KOST-Val weiter ausgewertet und differenziert ausgegeben.

SIP-Validierung: KOST-Val liest ein SIP (eCH-0160⁶ v1 und v1.1) und validiert die Muss-Punkte aus der SIP Spezifikation. Die einzelnen validierten Punkte werden in Gruppen wie z.B. Verzeichnisstruktur, Schema- und Prüfsummenvalidierung eingegliedert. Zuvor wird eine Formatvalidierung durchgeführt.

Die Resultate (inklusive Meldungen zu Inkonsistenzen oder Fehler) werden pro Schritt ausgegeben und in eine Validierungs-Logdatei geschrieben.

Die einzelnen Validierungsschritte / Prüfungen werden nacheinander ausgeführt. Wo möglich, wird die Validierung auch bei Fehlern weiter fortgesetzt, um die Anzahl von Korrekturzyklen zu reduzieren.

Detailliertere Informationen zu den einzelnen Formaten und Validierungsschritte sind im Anhang ersichtlich.

2 Systemvoraussetzungen

- Microsoft Windows 98 und neuer
- Mindestens 128 MB RAM (512 MB oder mehr empfohlen)
- Mindestens 20 GB Festplattenspeicher
- Java Runtime Environment (JRE) Version 8 respektive 1.8 inkl. JavaFX oder Liberica Full openJDK⁷

3 Offene Punkte / Feedback

Die offenen Punkte, von Bugs respektive Fehlern bis hin zu Ergänzungswünschen und Fragen, sind auf der Entwicklungsplattform GitHub unter Issues ersichtlich (<https://github.com/KOST-CECO/KOST-Val/issues>) und können an kost-val@kost-ceco.ch gemeldet werden.

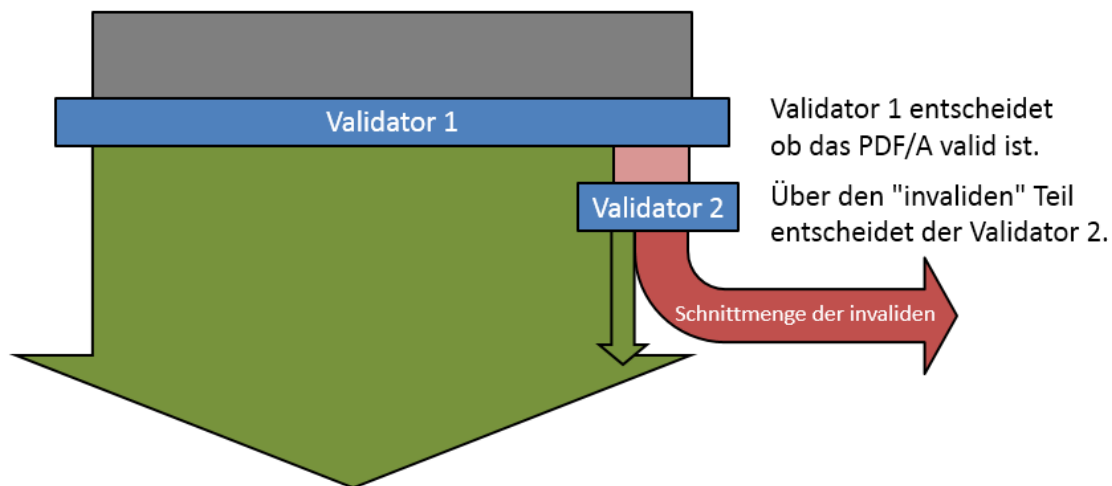
Diese Liste kann und soll durch jedermann erweitert werden und wird durch das Entwicklerteam bearbeitet.

⁶ Die Spezifikation kann von der eCH-Website heruntergeladen werden: <https://www.ech.ch/eCH/eCH-0160>.

⁷ Folgende Version wurde erfolgreich getestet: <https://download.bell-sw.com/java/14.0.2+13/bellsoft-jdk14.0.2+13-windows-amd64-full.msi>

4 Einleitung duale PDF/A-Validierung

Für PDF/A bietet KOST-Val die Möglichkeit einer dualen Validierung. Dabei wird eine PDF/A-Datei zunächst durch einen ersten Validator geprüft. Bei invalidem Resultat folgt eine Prüfung durch einen zweiten Validator. Die PDF/A-Datei gilt als valid, wenn mindestens einer der Validatoren sie als valid identifiziert, und als invalid, wenn beide Validatoren sie als invalid identifizieren.⁸



Die duale PDF/A-Validierung darf nur angewendet werden, wenn das Archiv es zulässt, dass potenziell invalide PDF/A-Dateien übernommen werden dürfen. Wenn dies nicht der Fall ist, dann sollte auf die duale PDF/A-Validierung verzichtet werden.

Für die duale Validierung wird sowohl 3-Heights™ PDF/A Validator von PDF-Tools als auch pdfaPilot von callas verwendet. Wenn nur ein Validator eingeschaltet ist, wird automatisch eine einfache Validierung durchgeführt.

Konzeptionelle Grundlage für die duale Validierung ist die Feststellung, dass selbst qualitativ hochstehende PDF/A-Validatoren zu unterschiedlichen Resultaten kommen können. Dies liegt einerseits daran, dass der eigentliche PDF/A-Standard ein Set von anderen Standards einschliesst, welche in den Validatoren nicht zwingend bis in alle Details implementiert sind. Andererseits sind gewisse Vorgaben des Standards so formuliert, dass sie legitimerweise auf verschiedene Arten implementiert werden können. Dass sämtliche relevanten Tools die Spezifikation einheitlich und vollständig implementieren, bleibt vorerst Zukunftsmusik. Deshalb bietet KOST-Val als Zwischenlösung die duale Validierung an.

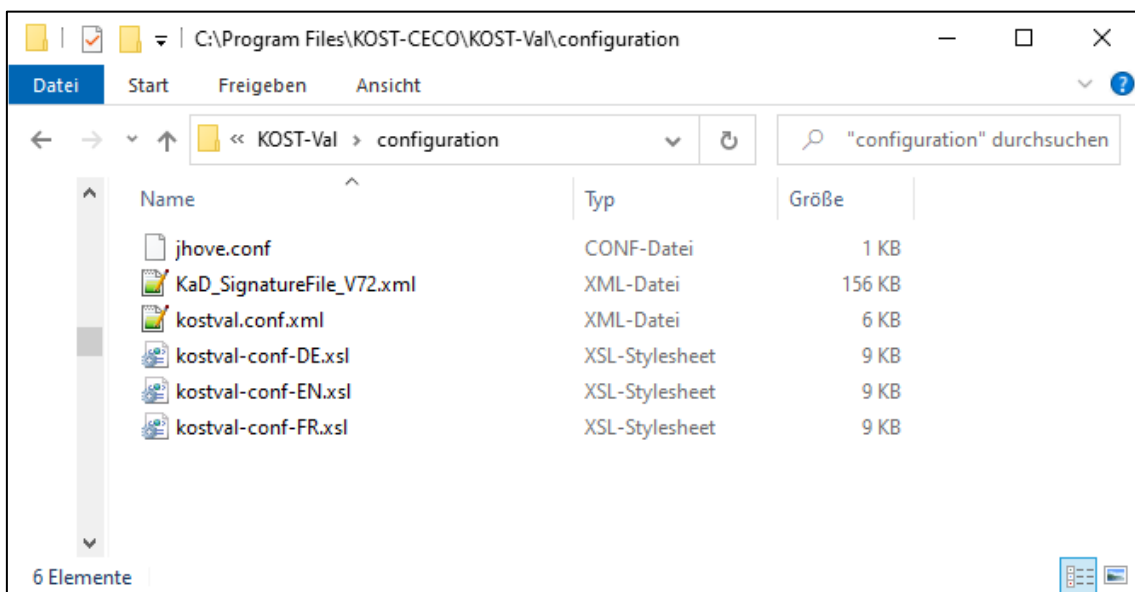
⁸ Die duale Validierung kann nur mit qualitativ hochstehenden PDF/A-Validatoren in diesem Sinne durchgeführt werden. Diese hohen Anforderungen erfüllen unter anderem die neuesten Versionen von 3-Heights™ PDF/A Validator von PDF-Tools und pdfaPilot von callas.

5 Installation von KOST-Val

- 1 KOST-Val herunterladen und in das gewünschte Verzeichnis entpacken.

! Ab der KOST-Val Version 1.9.1 werden die dll-Dateien in der 64bit Version im KOST-Val-Verzeichnis beigelegt. Sollte nur eine 32bit JRE8 zur Verfügung stehen, müssen die dll-Dateien mit den Dateien aus dem Verzeichnis «resources\3-Heights_PDF-Validator_dll\Win32» ausgetauscht werden. Je nach Berechtigung erfolgt dies direkt durch KOST-Val, ansonsten muss von Hand die dll ausgetauscht werden.

6 Konfiguration von KOST-Val



Im Ordner «configuration» sind die Dateien «jhove.conf» und «KaD_SignatureFile_V72.xml» abgelegt. Diese sind im Kapitel 6.2 beschrieben.

Die Konfigurationsdatei «kostval.conf.xml» sowie die drei Stylesheets werden, falls nicht korrekt vorhanden, ins Verzeichnis «USERHOME/.kost-val_2x/configuration» kopiert. Sämtliche Konfigurationen des KOST-Val können via GUI vorgenommen werden.

6.1 Bestandteile der Konfigurationsdatei «kostval.conf.xml»

Die Konfigurationsdatei «kostval.conf.xml» ist in verschiedenen Teilen aufgebaut. Die ausgelieferte Konfiguration ermöglicht eine sofortige Validierung von PDF/A, TIFF, SIARD, JP2, JPEG und SIPs. Nachfolgend werden die Bestandteile kurz beschrieben.

KOST-Val - Configuration

☒ JPEG

☒ JPEG2000

☒ SIARD
 ☒ SIARD-1.0 (eCR-0165 v1)
 ☒ SIARD-2.1

☒ PDF/A
 ☒ PDF Tools
 ☒ Callas
 ☒ PDF/A-1a
 ☒ PDF/A-2a
 ☒ Font
 ☐ Image
 ☐ JBIG2
 ☒ details
 ☐ N-Entry
 ☒ PDF/A-1b
 ☒ PDF/A-2b
 ☒ Tolerant
 ☒ PDF/A-2u

☒ TIFF

Komprimierungsalgorithmus	Farbraum	Bits per Sample (pro Kanal)	Sonstiges
<input checked="" type="checkbox"/> Uncompressed	<input checked="" type="checkbox"/> WhiteIsZero	<input checked="" type="checkbox"/> Bps 1	<input checked="" type="checkbox"/> Multipage
<input checked="" type="checkbox"/> CCITT 1D	<input checked="" type="checkbox"/> BlackIsZero	<input type="checkbox"/> Bps 2	<input type="checkbox"/> Tiles
<input checked="" type="checkbox"/> T4/Group 3 Fax	<input checked="" type="checkbox"/> RGB	<input checked="" type="checkbox"/> Bps 4	<input type="checkbox"/> Size
<input checked="" type="checkbox"/> T6/Group 4 Fax	<input checked="" type="checkbox"/> RGB Palette	<input type="checkbox"/> Bps 16	
<input checked="" type="checkbox"/> PackBits	<input type="checkbox"/> CMYK	<input type="checkbox"/> Bps 32	
	<input type="checkbox"/> YCbCr		
	<input type="checkbox"/> CIE L*a*b*		

☒ eCR-0160

Pfadlänge	179
SIP Name	SIP_[1-2][0-9]{3}[0-1][0-9][0-3][0-9]_w{3}
PUID	TXT PDFA1 PDFA2 TIFF JP2 JPEG WAVE MP3 MP4 MJ2 MPEG2 CSV SIARD WARC

Arbeitsverzeichnis

Inputverzeichnis

KOST-Val

Konfiguration

Angabe, ob eine JPEG-Validierung stattfinden soll [yes]:	yes
Angabe, ob eine JP2-Validierung stattfinden soll [yes]:	yes
Angabe, ob eine SIARD-Validierung stattfinden soll [yes]:	yes
Angabe, welche SIARD Versionen erlaubt sind [1.0, 2.1]:	1.0 2.1
Angaben zur PDF/A-Validierung [yes]:	yes
Angabe, ob eine PDF/A-Validierung mit PDF Tools stattfinden soll [yes]:	yes
Angabe, ob seitens PDF Tools auch detaillierte Fehler in Englisch ausgegeben werden sollen [yes]:	yes
Angabe, ob eine PDF/A-Validierung mit callas stattfinden soll [yes]:	yes
Angabe, ob seitens callas ein Fehler (E) oder eine Warnung (W) ausgegeben werden soll, wenn der N-Eintrag nicht übereinstimmt [W]:	W
Angabe, welche PDF/A Versionen erlaubt sind [1A, 1B, 2A, 2B, 2U]:	1A 1B 2A 2B 2U
Angabe, ob eine Schrift-Validierung (Durchsuch- und Extrahierbarkeit) stattfinden soll [tolerant]:	tolerant
Angabe, ob eine Bild-Validierung (JPEG und JP2) stattfinden soll [no]:	no
Angabe, ob die JBIG2-Komprimierung erlaubt ist [yes]:	yes
Angabe, ob eine TIFF-Validierung stattfinden soll [yes]:	yes
Angabe, der erlaubten Komprimierungsalgorithmen [Uncompressed, CCITT 1D, T4/Group 3 Fax, T6/Group 4 Fax, LZW, PackBits]:	Uncompressed CCITT 1D T4/Group 3 Fax T6/Group 4 Fax LZW PackBits
Angabe, der erlaubten Farbräume [WhiteIsZero, BlackIsZero, RGB, RGB Palette]:	WhiteIsZero BlackIsZero RGB RGB Palette
Angabe, der erlaubten Bits per Sample [1, 4, 8, 16]:	1 4 8 16
Angabe, ob Multipage-TIFFs erlaubt sind oder nicht [yes]:	yes
Angabe, ob der Aufbau in Kacheln erlaubt ist oder nicht [no]:	no
Angabe, ob Dateigrößen von 1000MB (~1GB) und grösser erlaubt sind oder nicht [no]:	no
Angaben zur SIP-Validierung [yes]:	yes
Erlaubte maximale Anzahl Zeichen in Pfadlängen [179]:	179
Vorgaben zum Aufbau des SIP-Namens [SIP_[1-2][0-9]{3}[0-1][0-9][0-3][0-9]_w{3}]:	SIP_[1-2][0-9]{3}[0-1][0-9][0-3][0-9]_w{3}
Auflistung der erlaubten Dateiformate [TXT, PDFA1, PDFA2, TIFF, JP2, JPEG, WAVE, MP3, MP4, MJ2, CSV, SIARD, WARC]:	TXT PDFA1 PDFA2 TIFF JP2 JPEG WAVE MP3 MP4 MJ2 MPEG2 CSV SIARD WARC
Arbeitsverzeichnis []:	
Inputverzeichnis []:	

6.2 «KaD_SignatureFile_V72.xml» und «jhove.conf»

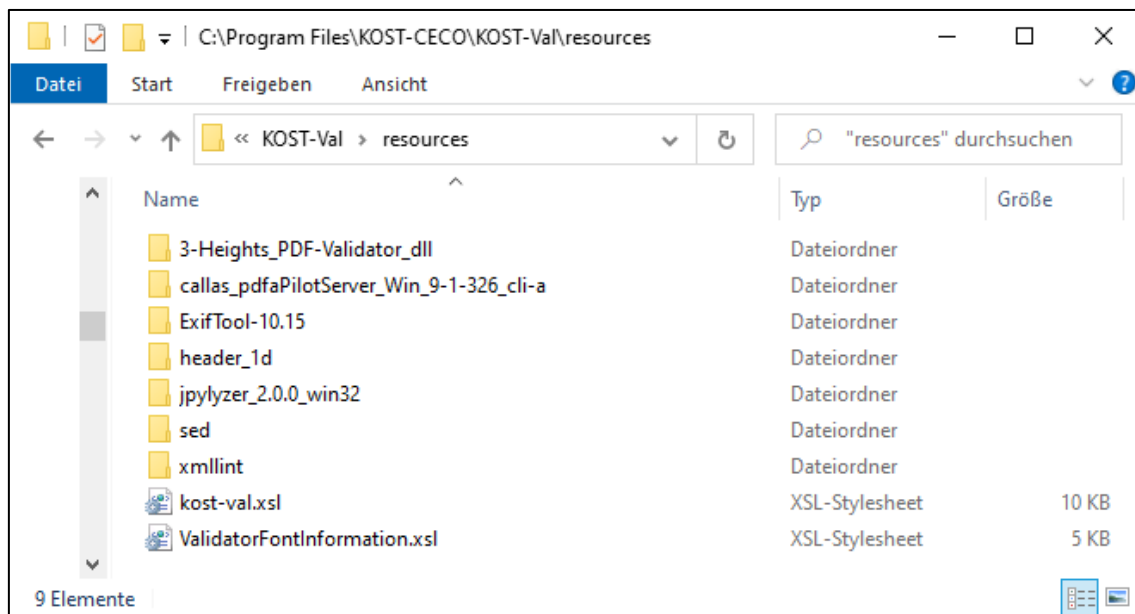
Im Ordner «configuration» sind die Dateien «KaD_SignatureFile_V72.xml» und «jhove.conf» abgelegt, welche nicht angepasst werden müssen.

«KaD_SignatureFile_V72.xml» wird für die Formaterkennung benötigt. Es enthält den Stand von DROID per 28.08.2013 und wurde zudem durch die KOST im Bereich der Erkennung der archivtauglichen Formate⁹ angepasst.

«jhove.conf» wird für die interne Jhove-Validierung benötigt.

7 Ressourcen von KOST-Val

Sämtliche Ressourcen von KOST-Val sind im Unterordner «resources» abgelegt.



⁹ Anstelle des SignatureFile von DROID verwendet KOST-Val das dazu kompatible KaD-SignatureFile der KOST (siehe auch https://github.com/KOST-CECO/KaD_SignatureFile). Dieses umfasst nur die im KaD untersuchten Formate und ermöglicht deren Erkennung in der von der KOST empfohlenen Granularität. Gegenüber dem DROID-SignatureFile werden damit eine signifikante Effizienzsteigerung und eine höhere Benutzerfreundlichkeit erreicht.

8 Validierung starten



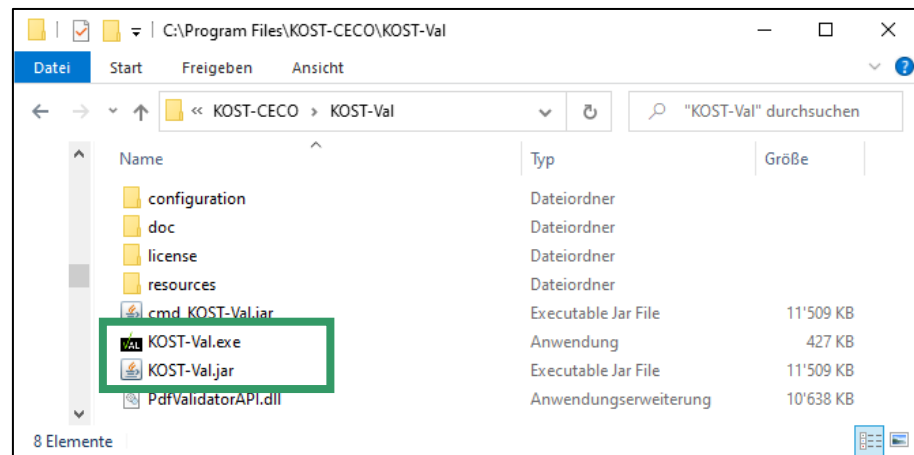
KOST-Val ist nicht Thread-sicher!

Das bedeutet, dass nicht mehrere Instanzen von KOST-Val gleichzeitig ausgeführt werden können, ohne sich gegenseitig zu behindern. Wird KOST-Val gleichzeitig ausgeführt, können Fehler wie z.B. eine fehlende Arbeitskopie vorkommen.

8.1 KOST-Val GUI

1

Starten von KOST-Val mittels Doppelklick auf «KOST-Val.exe»¹⁰ im Ordner «KOST-Val».



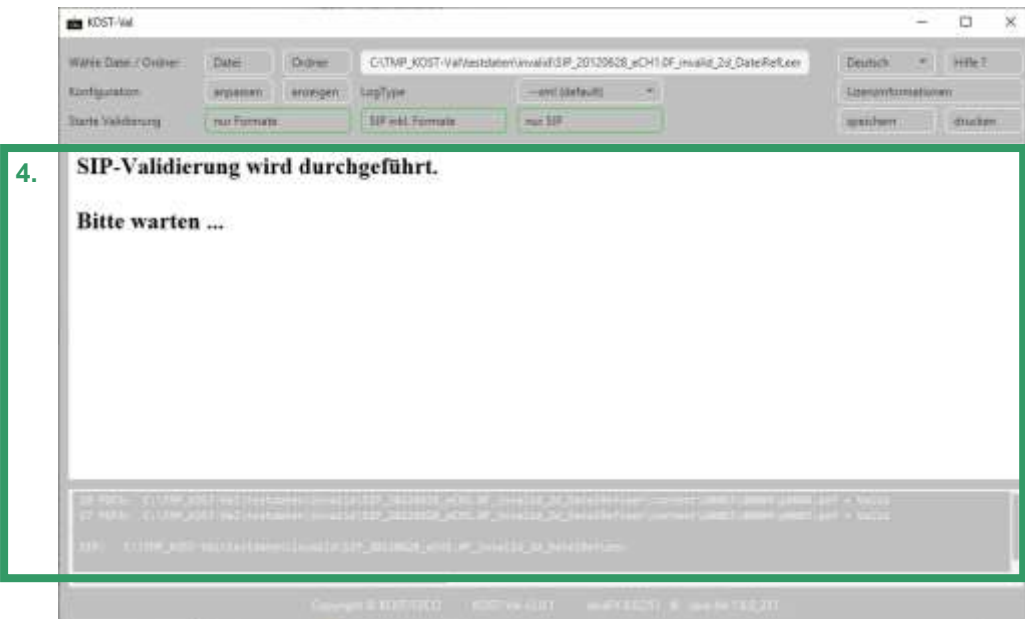
2

1. Datei oder Ordner zur Validierung angeben / auswählen
2. Ggf. Konfiguration und LogType anpassen
3. Validierung starten



¹⁰ Wird openJDK⁷ verwendet, muss das GUI via «KOST-Val.jar» und «Öffnen mit...» gestartet werden. Dabei muss das java.exe von openJDK angegeben werden.

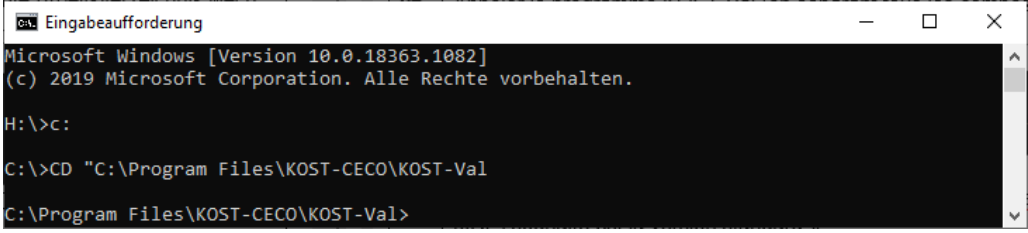
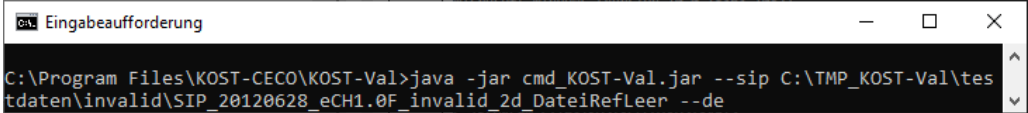
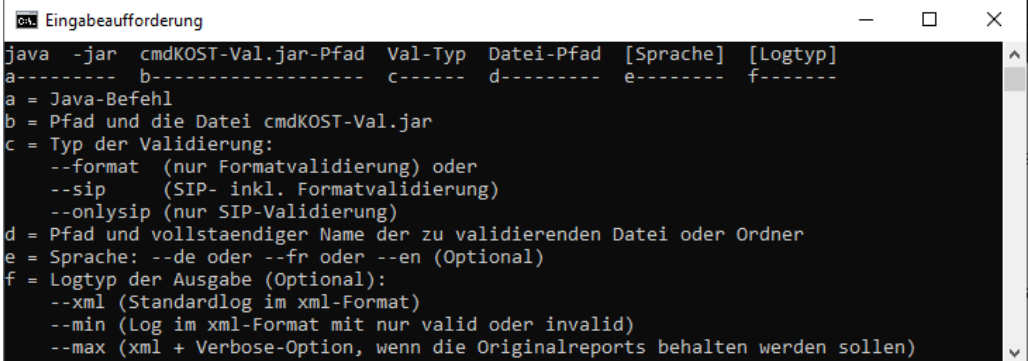
4. Warten bis die Validierung beendet ist.
Der Fortschritt / aktuelle Datei ist im unteren Feld ersichtlich.



- 3 Am Schluss der Validierung wird die log-Datei angezeigt. Diese führt zusätzliche Detailinformationen über die einzelnen invaliden Validierungsschritte auf, insbesondere der betroffene Validierungsschritt und der entsprechende Fehler. Wenn gewünscht kann die KOST-Val-Log-Datei gespeichert oder gedruckt werden.



8.2 Validierung manuell starten

1	<p>Eingabeaufforderung öffnen und in das gewünschte Arbeitsverzeichnis wechseln (CD "C:\Program Files\KOST-CECO\KOST-Val")¹¹.</p> 
2	<p>KOST-Val-Programmaufruf starten (die einzelnen Eingabebestandteile mit Leerzeichen trennen).</p> <pre>java -jar cmd_KOST-Val.jar --sip C:\TMP_KOST-Val\testdaten\invalid\SIP_20120628_eCH1.0F_invalid_2d_DateiRefLeer --de</pre>  <p><u>Anmerkungen:</u></p> <p>Die Eingabe <code>java -jar</code> ist nur möglich, wenn die gewünschte Java Runtime Environment (JRE) die Standardversion ist.</p> <p>Bei Bedarf kann die Einstellung des virtuellen Java Memory angepasst werden. <code>-Xmx</code> sollte bei «Out of Memory» und <code>-Xss</code> bei «Stack Overflow» Fehlern angepasst werden (<code>java -Xmx1024m -Xss128m -jar</code>).</p> <p>Wenn ein Eingabebestandteil Leerzeichen enthält, muss dieser in Anführungs- und Schlusszeichen eingegeben werden.</p> <p>KOST-Val kann auch von einem beliebigen Ort aus aufgerufen werden. Dies bedingt jedoch die Eingabe von absoluten Pfaden.</p> <p>Aufbau KOST-Val Befehl:</p> 

¹¹ Das Laufwerk wird z.B. mit `c:` gewechselt.

3

Die Datei wurde validiert, sobald "Valid" oder "Invalid" im cmd-Fenster erscheint. Der Ordner wurde validiert, sobald die Prompt (C:\Program Files\KOST-CECO\KOST-Val>) erscheint.

```

Eingabeaufforderung
17 PDF-A: C:\TMP_KOST-Val\testdaten\invalid\SIP_20120628_eCH1.0F_invalid_2d_DateiRefLeer\content\d0003\d0004\p0007.pdf = Valid

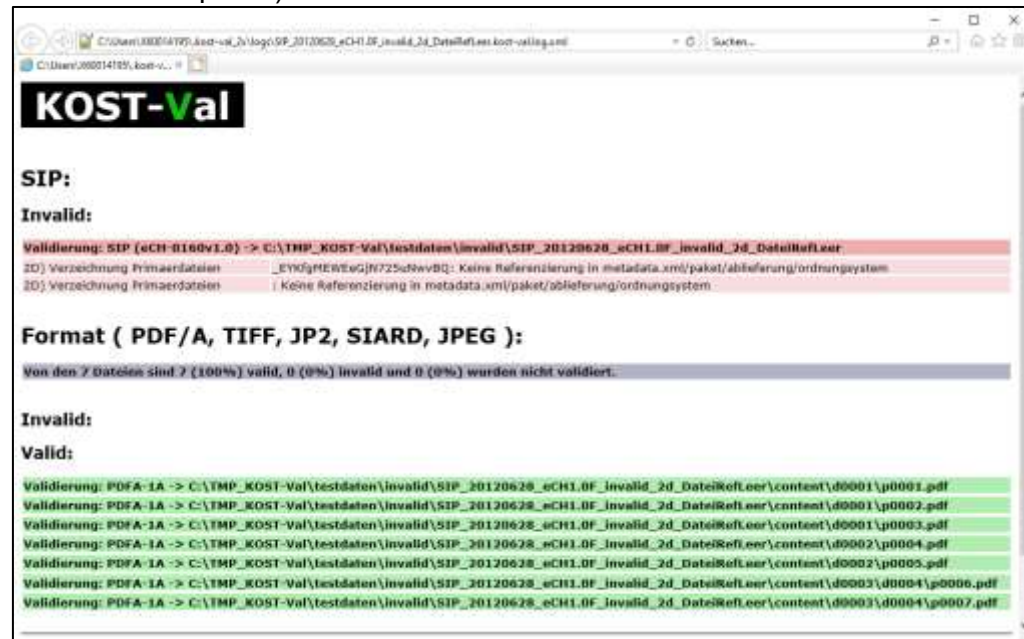
SIP: C:\TMP_KOST-Val\testdaten\invalid\SIP_20120628_eCH1.0F_invalid_2d_DateiRefLeer
Invalid

SIP-Validierung durchgefuehrt.
-> C:\Users\X60014195\.kost-val_2x\logs\SIP_20120628_eCH1.0F_invalid_2d_DateiRefLeer.kost-val.log.xml

C:\Program Files\KOST-CECO\KOST-Val>

```

Die detaillierten Resultate sind in der kost-val.log.xml-Datei ersichtlich (öffnen mit Internet Explorer).



Das Ergebnis der Gesamtvalidierung (korrekte/fehlerhafte Datei) wird ebenfalls ausgegeben und ist im *exit*-Status des Programms sichtbar, so dass die Validierung in eine automatisierte Verarbeitungskette eingebunden werden kann. Der *exit*-Status kann die folgenden Werte annehmen:

- 0 alles OK
- 1 Fehler im Programmaufruf
- 2 Validierung nicht bestanden

9 Beschreibung der Validierungsschritte

9.1 TIFF

Nachfolgend werden die einzelnen TIFF-Validierungsschritte detaillierter beschrieben.

A Erkennung

Wird die TIFF-Datei als TIFF erkannt?

B Jhove

Wurde die Jhove-Validierung bestanden?

C Komprimierung

Ist die verwendete Komprimierung gemäss der Konfiguration erlaubt?
Ist keine «Planar Configuration» enthalten?

D Farbraum

Ist der verwendete Farbraum gemäss der Konfiguration erlaubt?

E BitsPerSample

Sind die verwendeten «BitsPerSample» gemäss der Konfiguration erlaubt?

F Multipage

Entspricht die Verwendung der Eigenschaft «Multipage» der Konfiguration?

G Kacheln

Entspricht die Verwendung der Eigenschaft «Kacheln» der Konfiguration?

H Grösse

Ist die Dateigrösse von der Konfiguration erlaubt?

9.2 SIARD

Nachfolgend werden die einzelnen Validierungsschritte detaillierter beschrieben. Diese Kriterien sind ein Auszug der Muss-Kriterien aus der SIARD-Spezifikation.

- A Lesbarkeit**
Kann die SIARD-Datei gelesen werden?
- B primäre Verzeichnisstruktur**
Besteht eine korrekte primäre Verzeichnisstruktur?
- C Header-Validierung**
Ist der header-Ordner valid?
- D Struktur-Validierung**
Stimmt die Struktur aus metadata.xml mit der Datei-Struktur von content überein?
- E Spalten-Validierung**
Wurden die Angaben aus metadata.xml korrekt in die tableZ.xsd-Dateien übertragen?
- F Zeilen-Validierung**
Wurden die Angaben aus metadata.xml korrekt in die tableZ.xsd-Dateien übertragen?
- G Tabellen-Validierung**
Sind die Spaltennamen innerhalb der Tabelle einmalig?
- H Content-Validierung**
Sind die XML-Dateien im content valid zu ihrer Schema-Definition (XSD-Dateien)?
- I SIARD-Erkennung**
Wird die SIARD-Datei als SIARD erkannt?
- J Zusätzliche Primärdateien**
Sind alle im content-Ordner enthaltenen Dateien oder Ordner in metadata.xml beschrieben?
- W Warnung**
Wurden <dataOwner> und <dataOriginTimespan> ausgefüllt und nicht auf (...) belassen?

9.3 PDF/A

Die einzelnen Meldungen im Originalreport von callas sowie die Detailfehlermeldungen von PDF Tools werden anhand einzelner Wörter der Fehlermeldung den Modulen zugeordnet. Die Fehlermeldungen (Message) werden 1:1 übernommen. Bei der Validierung mit PDF Tools werden die Kategorien den Modulen zugeordnet, übersetzt und der Text ausgegeben.

A Allgemeines

Sind die allgemeinen Anforderungen wie z.B. nicht korrupte PDF/A-Datei erfüllt?

B Struktur

Ist die PDF/A-Datei korrekt aufgebaut (z.B. BOF und EOF)?

C Grafiken

Sind die Grafiken korrekt in der PDF/A-Datei integriert (z.B. valider Farbraum)? Stimmt die Komponentenanzahl im N-Eintrag des PDF/A OutputIntent mit ICC-Profil überein?¹²

D Schriften

Sind die Schriften korrekt in der PDF/A-Datei eingebettet (z.B. sind alle verwendeten Zeichen eingebettet)?

E Transparenz

Sind die Anforderungen betreffend Transparenz an die PDF/A-Datei erfüllt?

F Annotationen & Interaktionen

Enthält die PDF/A-Datei nur erlaubte Annotationen (z.B. keine 3D-Annotation)? Enthält die PDF/A-Datei nur erlaubte Interaktionen?

G Aktionen

Enthält die PDF/A-Datei nur erlaubte Aktionen (z.B. kein JavaScript)?

H Metadaten

Sind die Metadaten korrekt in der PDF/A-Datei enthalten?

I Zugänglichkeit

Sind die Anforderungen an die Zugänglichkeit (Conformance Level A) erfüllt?

J JBIG2-Kontrolle (konfigurierbar)

Wurde die JBIG2-Komprimierung verwendet?

K Schrift-Validierung (konfigurierbar)

Sind die enthaltenen Schriften durchsuch- und extrahierbar?

¹² Dazu wird bei pdfaPilot von callas eine zusätzliche Prüfung durchgeführt, welche je nach Konfiguration (siehe 6.1) ggf. eine Warnung oder einen Fehler ausgibt. Bei 3-Heights™ PDF/A Validator von PDF Tools wird diese Prüfung standardmässig durchgeführt, jedoch in der iCategory_1 bemängelt, welches aufgrund der Zusammensetzung unter A) Allgemeines ausgegeben wird.

9.4 JP2

Die nicht bestandenen Tests von Jpylyzer werden den Modulen zugeordnet, und entsprechende Fehlermeldungen werden ausgegeben. Nachfolgend werden die einzelnen JP2-Validierungsschritte detaillierter beschrieben.

A Erkennung und Jpylyzer

Wird die JP2-Datei als JP2 erkannt? Wurde die Jpylyzer-Validierung bestanden?

B Metadaten-Validierung

Wurden die Metadaten korrekt erfasst?

C Bild-Validierung

Ist das Bild korrekt aufgebaut?

D Erweiterte Validierung

Wurden die zusätzlichen Jpylyzer-Tests bestanden?

9.5 JPEG

Die Fehlermeldungen aus BadPeggy werden den Modulen zugeordnet, und übersetzt in Deutsch ausgegeben (Original in Englisch). Nachfolgend werden die einzelnen JPEG-Validierungsschritte detaillierter beschrieben.

A Erkennung und BadPeggy

Wird die JPEG-Datei als JPEG erkannt? Wurde die BadPeggy-Validierung bestanden?

B Korrupte Daten

Gibt BadPeggy eine Fehlermeldung heraus, welche auf korrupte Daten hinweist?

C Ungültige Dateistruktur

Gibt BadPeggy eine Fehlermeldung heraus, welche auf eine ungültige Dateistruktur hinweist?

D Andere Probleme

Gibt BadPeggy eine Fehlermeldung aus, welche noch nicht den Modulen zugeordnet und übersetzt wurde?

9.6 SIP

Nachfolgend werden die einzelnen Validierungsschritte detaillierter beschrieben. Diese Kriterien sind ein Auszug der Muss-Kriterien aus der SIP-Spezifikation.

Modul 1: Paket- und XML-Konsistenz

1A Lesbarkeit

Kann das SIP (ZIP / ZIP64) fehlerfrei geöffnet werden?

1B primäre Verzeichnisstruktur

Besteht eine korrekte primäre Verzeichnisstruktur?

1C Verzeichnis- und Dateinamen

Entsprechen die Namen den Einschränkungen in der Spezifikation?

1D Schemavalidierung metadata.xml

Entspricht metadata.xml den Schemadateien in /header/xsd?

1E SIP-Typ ermitteln

Der SIP Typ wird ermittelt und angezeigt: GEVER oder FILE

1F Primärdateien im Verzeichnis

Sind Primärdateien im Verzeichnis /content vorhanden?¹³

Modul 2: Datei-Konsistenz

2A Fehlende Primärdateien

Sind alle referenzierten Dateien vorhanden?

2B Zusätzliche Primärdateien

Sind im SIP keine zusätzlichen Primärdateien vorhanden?

2C Prüfsummen-Validierung

Stimmen die Prüfsummen der Dateien mit Prüfsumme überein?

2D Verzeichnung Primärdateien

Sind alle referenzierten Dateien auch im Ordnungssystem verzeichnet?

Modul 3: Dateiformat- und Datums-Konsistenz

3A Formaterkennung

Sind die erkannten Formate erlaubt?

3B Zusätzliche Formate

Alle Dateien in nicht erlaubten Formaten mit entsprechenden Formatangaben auflisten.

3C Formatvalidierung

Sind die Formate valid?

3D Zeitraum-Validierung

Stimmen die Zeitangaben in (metadata.xml)/ablieferung überein?

¹³ Primärdateien können in einem GEVER-SIP fehlen, das nur zur Archivierung einer Ordnerstruktur dient, aber nicht in einem FILE-SIP.

10 Urheberrecht

KOST-Val ist eine Entwicklung der KOST. Alle Rechte liegen bei der KOST. KOST-Val wurde im 2012 durch die KOST unter der GNU General Public License v3+ veröffentlicht.

Notice:	This product includes software developed by the Apache Software Foundation (http://www.apache.org/).
----------------	---

KOST-Val stützt sich auf folgende unveränderte Komponenten anderer Hersteller, welche direkt im Quellcode von KOST-Val eingebunden sind:

Drittprogramm / -Komponente	Version	Lizenz
3-Heights™ PDF/A Validator API http://www.pdf-tools.com	6.11.0.7	Siehe Kapitel 10.1
Apache Commons http://commons.apache.org/ - commons-logging-1.2.jar - commons-io-2.6.jar	1.2 2.6	Apache License 2.0
Apache log4j http://logging.apache.org/log4j/	1.2.12	Apache License 2.0
Apache Xerces http://xerces.apache.org/	2.7.1	Apache License 2.0
BadPeggy http://coderslagoon.com/	2.0	GPL v3 License
DROID http://digital-preservation.github.io/droid/	5.0.3	3c BSD- License
Jdom 2.0.0 http://www.jdom.org/	2.0.0	jdom License
Jhove http://hul.harvard.edu/jhove/	1.5	LGPL v2.1 License
Junit 4.12 http://www.junit.org/	4.12	CPL v1.0
Spring Framework API http://static.springsource.org/spring/docs/5.0.x/api/	5.0.8	Apache License 2.0
zip64 http://sourceforge.net/projects/zip64file/	1.02	GPL v2+ License

KOST-Val stützt sich auf folgende unveränderte Komponenten anderer Hersteller, welche mit KOST-Val abgegeben werden:

Drittprogramm / -Komponente	Version	Lizenz
ExifTool http://www.sno.phy.queensu.ca/~phil/exiftool/	10.15	PERL respektive GPL v3.0 License
Jpylyzer http://jpylyzer.openpreservation.org/	2.0.0	LGPL v3.0 License
pdfaPilot CLI https://www.callassoftware.com	9.1.326	Siehe Kapitel 10.2
GNU sed https://www.gnu.org/software/sed	4.4	GPL v3+ License
Xmlint http://xmlsoft.org/xmlint.html/	20630	MIT License

Die Benutzer von KOST-Val sind gehalten, die Lizenzbestimmungen all dieser Komponenten zu befolgen, welche im Verzeichnis KOST-Val\license vorliegen.

10.1 3-Heights™ PDF/A Validator API-Lizenz

Für die Verwendung der Eingeschränkten Version des 3-Heights™ PDF/A Validator von PDF Tools hat die KOST folgende Individuelle Vereinbarung zu den Allgemeinen Lizenzbedingungen mit PDF Tools vereinbart:

2. Individuelle Vereinbarung

Dieses Vertragsverhältnis regelt die Client-Lizenz zwischen der PDF TOOLS als Lizenzgeber und der KOST als Lizenznehmer gemäss nachfolgenden Spezialbestimmungen:

- PDF Tools AG erteilt für KOST eine kostenfreie OEM-Lizenz für das 3-Heights™ PDF/A Validator API als Zusatzfunktion ihrer eigenen Validator-Software (KOST-Val).
- Die Lizenz schliesst den Gebrauch der Software (KOST-Val) durch Gedächtnisinstitutionen, bestehend aus Archiven oder Bibliotheken, deren Zulieferer und der KOST selbst, ein.
- Der OEM-Lizenzschlüssel, welcher fest in KOST-Val eingebunden ist, darf nicht ausserhalb der Applikation (KOST-Val) verwendet werden.
- Die Lizenz ist zeitlich unbegrenzt, jedoch bezüglich Durchsatz pro Installation begrenzt (72'000 Seiten pro Jahr).
- Für die Verteilung der Software (KOST-Val) an den Anwender ist die KOST zuständig.
- Der First Level Support der Anwender erfolgt durch KOST. Second Level Support Fälle leitet KOST an PDF Tools AG weiter.
- Wenn der Anwender weitergehende Bedürfnisse hat, z.B. höherer Durchsatz, Integration in andere Applikationen etc. kauft er die Software (3-Heights™ PDF/A Validator API) direkt bei PDF Tools AG.
- Die KOST darf weiterhin den Quellcode von KOST-Val Open Source publizieren und KOST-Val gratis und ohne Registrierung abgeben.

Für die Benutzer sind folgende Punkte massgebend:

- Die Lizenz schliesst den Gebrauch der Software (KOST-Val) durch Gedächtnisinstitutionen, bestehend aus Archiven oder Bibliotheken, deren Zulieferer und der KOST selbst, ein.
- Der OEM-Lizenzschlüssel, welcher fest in KOST-Val eingebunden ist, darf nicht ausserhalb der Applikation (KOST-Val) verwendet werden.
- Die Lizenz ist zeitlich unbegrenzt, jedoch bezüglich Durchsatz pro Installation begrenzt (72'000 Seiten pro Jahr).
- Der First Level Support der Anwender erfolgt durch KOST. Second Level Support Fälle leitet KOST an PDF Tools AG weiter.
- Wenn der Anwender weitergehende Bedürfnisse hat, z.B. höherer Durchsatz, Integration in andere Applikationen etc. kauft er die Software (3-Heights™ PDF/A Validator API) direkt bei PDF Tools AG.

Die Benutzer von KOST-Val sind gehalten, diese Lizenzbestimmung zu befolgen.

10.2 pdfaPilot CLI Lizenz

Für die Verwendung der Eingeschränkten Version des pdfaPilot CLI von callas hat die KOST folgende Individuelle Vereinbarung zu den Allgemeinen Lizenzbedingungen mit callas vereinbart:

2. Individuelle Vereinbarung

Dieses Vertragsverhältnis regelt die Lizenz zwischen der callas software als Lizenzgeber und der KOST als Lizenznehmer gemäss nachfolgenden Spezialbestimmungen:

- callas software erteilt für die KOST eine kostenfreie Lizenz für callas pdfaPilot CLI für Windows zur innerbetrieblichen Nutzung und zur Integration in ihren eigenen Validator „KOST-Val“.
- Die Lizenz schliesst die Distribution von KOST-Val an „Anwender“ (Gedächtnisinstitutionen, Archive oder Bibliotheken und deren Zulieferer) ein.
- Für die Distribution von KOST-Val an diese Anwender ist die KOST zuständig und darf KOST-Val auch gratis und ohne Registrierung an diese abgeben.
- Die Lizenz ist zeitlich unbegrenzt, jedoch bezüglich Durchsatz pro Installation begrenzt auf 72'000 Seiten pro Jahr.
- Die KOST darf den eigenen Quellcode von KOST-Val Open Source publizieren. callas pdfaPilot CLI ist hiervon ausgenommen.
- First Level Support der Anwender erfolgt durch die KOST. Second Level Support leistet callas software gegenüber der KOST.

Für die Benutzer sind folgende Punkte massgebend:

- Die Lizenz schliesst die Distribution von KOST-Val an «Anwender» (Gedächtnisinstitutionen, Archive oder Bibliotheken und deren Zulieferer) ein.
- Die Lizenz ist zeitlich unbegrenzt, jedoch bezüglich Durchsatz pro Installation begrenzt auf 72'000 Seiten pro Jahr.
- Die KOST darf den eigenen Quellcode von KOST-Val Open Source publizieren. callas pdfaPilot CLI ist hiervon ausgenommen.
- First Level Support der Anwender erfolgt durch die KOST. Second Level Support leistet callas software gegenüber der KOST.

Die Benutzer von KOST-Val sind gehalten, diese Lizenzbestimmung zu befolgen.

11 Anhang

11.1 Programmaufbau

KOST-Val wurde nachfolgenden Anforderungen aufgebaut:

Funktionale Anforderungen:

TIFF-Validierung: KOST-Val liest ein TIFF und validiert mit Hilfe von Jhove die folgenden Punkte:

Validierungsschritt	Bezeichnung
A (Abbruch wenn Fehler)	Erkennung
B	Jhove
C	Komprimierung
D	Farbraum
E	BitsPerSample
F	Multipage
G	Kacheln
H	Grösse

SIARD-Validierung: KOST-Val liest ein SIARD und validiert die folgenden Punkte:

Validierungsschritt	Bezeichnung
A (Abbruch wenn Fehler)	Lesbarkeit
B (Abbruch wenn Fehler)	Primäre Verzeichnisstruktur
C (Abbruch wenn Fehler)	Header-Validierung
D (Abbruch wenn Fehler)	Struktur-Validierung
E	Spalten-Validierung
F	Zeilen-Validierung
G	Tabellen-Validierung
H	Content-Validierung
I	SIARD-Erkennung
J	Zusätzliche Primärdateien
W	Warnung

PDF/A-Validierung: KOST-Val liest ein PDF und validiert mit Hilfe des 3-Heights™ PDF/A Validator von PDF-Tools oder auch des pdfaPilot von callas die folgenden Punkte:

Validierungsschritt	Bezeichnung
A (Abbruch wenn Fehler)	Allgemeines
B	Struktur
C	Grafiken
D	Schriften
E	Transparenz
F	Annotationen
G	Aktionen & Interaktionen
H	Metadaten
I	Zugänglichkeit
J (konfigurierbar)	JBIG2-Kontrolle
K (konfigurierbar)	Schrift-Validierung

JP2-Validierung: KOST-Val liest ein JP2 und validiert mit Hilfe von Jpylyzer die folgenden Punkte:

Validierungsschritt	Bezeichnung
A (Abbruch möglich wenn Fehler)	Erkennung und Jpylyzer
B	Metadaten-Validierung
C	Bild-Validierung
D	Erweiterte Validierung

JPEG-Validierung: KOST-Val liest ein JPEG und validiert mit Hilfe von BadPeggy die folgenden Punkte:

Validierungsschritt	Bezeichnung
A (Abbruch möglich wenn Fehler)	Erkennung und BadPeggy
B	Korrupte Daten
C	Ungültige Dateistruktur
D	Andere Probleme

SIP-Validierung: KOST-Val liest ein SIP und validiert die folgenden Punkte aus der SIP Spezifikation:

Validierungsschritt	Bezeichnung (Stepname)
1a (Abbruch wenn Fehler)	Lesbarkeit
1b (Abbruch wenn Fehler)	primäre Verzeichnisstruktur
1c (Abbruch wenn Fehler)	Verzeichnis- und Dateinamen
1d (Abbruch wenn Fehler)	Schemavalidierung metadata.xml
1e	SIP-Typ ermitteln
1f	Primärdateien im Verzeichnis
2a	Fehlende Primärdateien
2b	Zusätzliche Primärdateien
2c	Prüfsummen-Validierung
2d	Verzeichnung Primärdateien
3a	Formaterkennung
3b	Zusätzliche Formate
3c	Formatvalidierung
3d	Zeitraum-Validierung

Die Resultate (inklusive Meldungen zu Inkonsistenzen oder Fehler) werden pro Schritt ausgegeben und in eine Validierungs-Logdatei geschrieben.

Das Ergebnis der Gesamtvalidierung (korrekte/fehlerhafte Datei) wird ebenfalls ausgegeben und im *exit*-Status des Programms sichtbar, so dass die Validierung in eine automatisierte Verarbeitungskette eingebunden werden kann. Der *exit*-Status kann die folgenden Werte annehmen:

- 0 alles OK
- 1 Fehler im Programmaufruf
- 2 Validierung nicht bestanden

Die einzelnen Validierungsschritte / Prüfungen werden nacheinander ausgeführt. Wo möglich, wird die Validierung auch bei Fehlern weiter fortgesetzt, um die Anzahl von Korrekturzyklen zu reduzieren.

Nichtfunktionale Anforderungen:

Für besondere Aufgaben werden externe Programme oder entsprechende Java-Frameworks eingesetzt.

Die Anwendung ist modular aufgebaut, damit ohne viel Aufwand ein oder mehrere weitere Validierungsmodule eingebaut werden können.

Die Log-/Programmausgabe erlaubt ein einfaches Auslesen des Ergebnisses der einzelnen Validierung und damit die Verwendung des Tools in einer Prozesskette,

Die Konsolenausgabe begrenzt sich auf die Bezeichnung der Validierungsart, das Gesamtergebnis "valid" oder "invalid" sowie der Pfad zur Datei. Alle zusätzlichen Informationen werden in der Log-Datei aufgeführt.

11.2 Funktionsprinzip Formatvalidierung

