

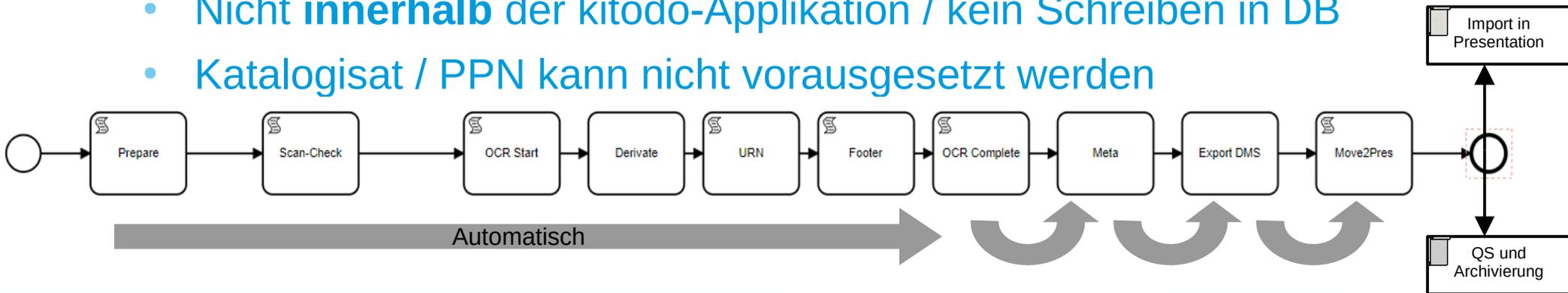
Realisierung eines kitodo-production-Workflows auf Basis einfacher Shellscripte

ULB Darmstadt / Voraussetzungen

- Seit 2008 Digitalisierung in der ULB Darmstadt
- Plattform: dwork (UB Heidelberg)
- Ca. 15.000 Objekte aller Art
- Strategische Entscheidung für kitodo
 - Gründe u.a.: Community, Lizenz, Support, Entwicklung – Features (?)
- Kaum valide Titelaufnahmen
- Enge personelle Kapazitäten
- Hessische Parlamentsprotokolle (MDL, Teil 2) als Pilot

Infrastruktur

- kitodo.production:
 - nur „Derivate“, „Meta“ und „Export DMS“
- → andere „Funktionsgruppen“  müssen selbst realisiert werden
- Aufgrund unserer Voraussetzungen:
 - **Einfach**, auf Basis von Shellscripten
 - Nicht **innerhalb** der kitodo-Applikation / kein Schreiben in DB
 - Katalogisat / PPN kann nicht vorausgesetzt werden



Funktionsgruppen

- Prepare
- Scan-Check (Audit)
- OCR (Start und Complete)
- URN (und Meta)
- Footer
- Move to Presentation
- Import in Presentation
- QS und Archivierung

Realisierung 1

- Anbindung HEBIS-Verbund
- Kleine Funktionsbibliothek, einfache Konfigurationsdatei(en)
- Immer identischer Aufbau der bash-Scripts
- Anpassung an Struktur von kitodo.production
 - Projekte (1 Projekt → 1 Config-Datei)
 - Shellscripte als TaskScripts im Workflow von kitodo
 - Scripte schreiben in / lesen aus meta.xml
- Tickets zur Zustandskontrolle der Tasks / Workflow-Phasen
- Funktionen, die nicht in kitodo-GUI möglich sind → **kitotick**

Realisierung 2

- Start Frühjahr 2021
- 3 github-Repos (Taskscripts, Config-Dateien, kitotick)
- Ca. 50 Scripts mit ca. 6000 Zeilen Code
- Bisher ca. 100 Bände Hessische Parlamentsprotokolle verarbeitet
- Mittlerweile v 3.3 von kitodo.production
- kitodo.presentation mit Templates von Zeutschel
- ...

```
1 # Add footer with logo and text to XSL for a given ProcessID
2 #
3 # NAME: ProcessID (StepID)
4 # PARAMS: ProcessID (StepID)
5 #
6 #
7 #
8 #
9 #
10 #
11 #
12 #
13 #
14 #
15 #
16 #
17 #
18 #
19 #
20 #
21 #
22 #
23 #
24 #
25 #
26 #
27 #
28 #
29 #
30 #
31 #
32 #
33 #
34 #
35 #
36 #
37 #
38 #
39 #
40 #
41 #
42 #
43 #
44 #
45 #
46 #
47 #
48 #
49 #
50 #
51 #
52 #
53 #
54 #
55 #
56 #
57 #
58 #
59 #
60 #
61 #
62 #
63 #
64 #
65 #
66 #
67 #
68 #
69 #
70 #
71 #
72 #
73 #
74 #
75 #
76 #
77 #
78 #
79 #
80 #
81 #
82 #
83 #
84 #
85 #
86 #
87 #
88 #
89 #
90 #
91 #
92 #
93 #
```

Todos / Lessons learned / Caveats

- Performanz und Stabilität bisher positiv zu beurteilen
- Support von Community ist gut, aber nicht so belastbar, dass man als Neuling ohne "bezahlte" Hilfestellung zum Erfolg kommt
- es wird erhebliches Knowhow benötigt, um produktiv zu werden:
 - Systemadministration
 - METS/MODS
 - XML/XSLT/XPATH etc.
 - Scripting / Coding
 - Bibliothekarische Formate (Mapping Pica -> kitodo-METS/MODS)
 - Workflow-Modellierung
 - github / agiles Entwickeln
- Den Einstieg erleichtern würde eine schnell und stabil einsetzbare default-Konfiguration der rulesets

Realisierung eines kitodo-production-Workflows auf Basis einfacher Shellscripte

23.11.2021

Roland Roth-Steiner, ULB Darmstadt

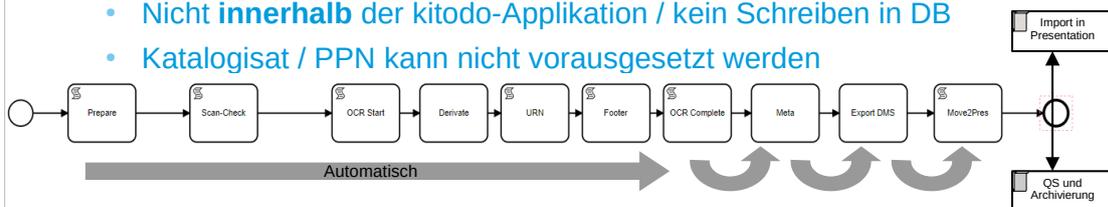
1

ULB Darmstadt / Voraussetzungen

- Seit 2008 Digitalisierung in der ULB Darmstadt
- Plattform: dwork (UB Heidelberg)
- Ca. 15.000 Objekte aller Art
- Strategische Entscheidung für kitodo
 - Gründe u.a.: Community, Lizenz, Support, Entwicklung – Features (?)
- Kaum valide Titelaufnahmen
- Enge personelle Kapazitäten
- Hessische Parlamentsprotokolle (MDL, Teil 2) als Pilot

Infrastruktur

- kitodo.production:
 - nur „Derivate“, „Meta“ und „Export DMS“
- → andere „Funktionsgruppen“ müssen selbst realisiert werden
- Aufgrund unserer Voraussetzungen:
 - **Einfach**, auf Basis von Shellscripten
 - Nicht **innerhalb** der kitodo-Applikation / kein Schreiben in DB
 - Katalogisat / PPN kann nicht vorausgesetzt werden



Funktionsgruppen

- Prepare
- Scan-Check (Audit)
- OCR (Start und Complete)
- URN (und Meta)
- Footer
- Move to Presentation
- Import in Presentation
- QS und Archivierung

Realisierung 1

- Anbindung HEBIS-Verbund
- Kleine Funktionsbibliothek, einfache Konfigurationsdatei(en)
- Immer identischer Aufbau der bash-Scripts
- Anpassung an Struktur von kitodo.production
 - Projekte (1 Projekt → 1 Config-Datei)
 - Shellscripte als TaskScripts im Workflow von kitodo
 - Scripte schreiben in / lesen aus meta.xml
- Tickets zur Zustandskontrolle der Tasks / Workflow-Phasen
- Funktionen, die nicht in kitodo-GUI möglich sind → **kitotick**

Realisierung 2

- Start Frühjahr 2021
- 3 github-Repos (Taskscripts, Config-Dateien, kitotick)
- Ca. 50 Scripts mit ca. 6000 Zeilen Code
- Bisher ca. 100 Bände Hessische Parlamentsprotokolle verarbeitet
- Mittlerweile v 3.3 von kitodo.production
- kitodo.presentation mit Templates von Zeutschel
- ...

A screenshot of a code editor displaying a script. The code is written in a light-colored font on a dark background. It includes various annotations, comments, and code blocks, likely related to the processing of historical documents as mentioned in the text. The code appears to be a mix of shell scripts and configuration files.

Todos / Lessons learned / Caveats

- Performanz und Stabilität bisher positiv zu beurteilen
- Support von Community ist gut, aber nicht so belastbar, dass man als Neuling ohne "bezahlte" Hilfestellung zum Erfolg kommt
- es wird erhebliches Knowhow benötigt, um produktiv zu werden:
 - Systemadministration
 - METS/MODS
 - XML/XSLT/XPATH etc.
 - Scripting / Coding
 - Bibliothekarische Formate (Mapping Pica -> kitodo-METS/MODS)
 - Workflow-Modellierung
 - github / agiles Entwickeln
- Den Einstieg erleichtern würde eine schnell und stabil einsetzbare default-Konfiguration der rulesets