Automatisiertes Publishing aus TYPO3

Jan Oevermann

T2B711 Informationssysteme B
Hochschule Karlsruhe – Technik und Wirtschaft

Inhalt

1 Einleitung

Während meines Praxissemesters bei der Projektron GmbH in Berlin hatte ich die Möglichkeit, mich mit der Evaluierung und Entwicklung einer automatisierten Publikationsstrecke aus TYPO3 in PDF zu beschäftigen. Die vorliegende Arbeit umfasst allgemeine Anforderungen, evaluierte Lösungsansätzen und die von mir entwickelte Umsetzung. Sowohl für die Ausgabe eines verwertbaren Formats aus TYPO3 als auch für die anschließende Verarbeitung zu einem PDF mussten Lösungen gefunden werden. Obwohl die fertige Publikationsstrecke auf speziellen Anforderungen und einem eigenen Informationsmodell beruht, kann die grundlegende Vorgehensweise auch auf andere Anwendungsfälle übertragen werden.

2 TYPO3

Das freie Web Content Management System TYPO3 (Eigenbezeichnung: TYPO3 - Open Source Enterprise Content Management System[1]) hat sich seit seiner Erscheinung 1998 zu den am meisten verbreiteten Open-Source-CMS entwickelt[2]. Aufgrund seiner zahlreichen Erweiterungen (im Typo3-Sprachgebrauch 'Extensions'), seiner skalierbaren Architektur und hohen Scriptfähigkeiten wächst die Beliebtheit des Systems deswegen nicht nur im privaten sondern auch im Enterprise-Bereich weiter. Typo3 besitzt eine eigene Templatingsprache – TypoScript - mit deren Hilfe sich Inhalte mit ihren entsprechenden Darstellungselementen verknüpfen lassen. Mehrsprachigkeit und Rechtevergabe sind tief in das System integriert, ebenso wie die Möglichkeit ein Digital Asset Management (DAM)[3] als Extension zu integrieren, was auch den Einsatz in der Technischen Redaktion interessant macht.

3 Ausgangssituation

3.1 Relaunch der Website mit Typo3

Die Projektron GmbH mit Sitz in Berlin konnte Ende des Jahres 2011 den erfolgreichen Relaunch der Firmen-Website auf Basis von Typo3 feiern. Die technische Umsetzung, Template- und Extension-Programmierung sowie Anbindungen an hausinterne Systeme erfolgte in enger Zusammenarbeit mit der Hannoveraner Agentur ID.on[4]. Im Zuge des Relaunchs wurden auch die kompletten produktbeschreibenden Inhalte erneuert und umfangreich erweitert.
Diese sollen auch weiterhin (ausschließlich) in Typo3 gepflegt und organisiert werden. Zugriff auf das System haben neben Technischen Redakteure auch Mitarbeiter aus Marketing, Verwaltung und Vertrieb. Rechte und Rollen werden in Typo3 verwaltet.

---

[1] http://typo3.org/ - Im weiteren Verlauf der besseren Lesbarkeit wegen: Typo3

[2] http://w3techs.com/technologies/segmentation/cl-de-/content_management

[3] http://typo3.org/extensions/repository/view/dam

[4] http://id-on.de/

3.2 Produktbeschreibung

Die bisherige Produktbeschreibung, die in der Regel als PDF an Interessenten verteilt oder gedruckt auf Veranstaltungen zur Verfügung gestellt wird, wurde bisher im Redaktionssystem TCToolbox von Ovidius gepflegt und ausgespielt. Da nun die Inhalte aus der Produktbeschreibung nahezu deckungsgleich mit denen der Website waren, wurde eine Vereinheitlichung der beiden Quellen angestrebt - die neue Produktbeschreibung sollte nun direkt aus Typo3 in PDF publiziert werden können.

Folgende wirtschaftliche und organisatorische Vorteile ergeben sich aus der Vereinheitlichung der beiden Publikationsstränge:

- Reduzierung der Übersetzungskosten für 4 Zielsprachen
- Schnellere Veröffentlichung durch Vereinfachung des Aktualisierungsvorgangs
- Bessere Qualitätssicherung durch die Vermeidung doppelter Datenhaltung
- Einsparung von Ressourcen durch eine Automatisierung der Publikation

Da die Produktbeschreibung mindestens mit jedem Kunden-Release der Software aktualisiert werden muss, multiplizieren[5] sich die nachteiligen Kostenfaktoren der bisherigen Vorgehensweise und rechtfertigen Evaluierung und Entwicklung einer neuen Lösung.

4 Anforderungen

Die speziellen Anforderungen, die die Projektron GmbH an die neue Publikationsstrecke stellte, entsprechen weitgehend den generellen Anforderungen an ein modernes automatisiertes Cross-Media-Publishing. Sie sind im Folgenden aufgelistet:

4.1 Integration

Die Publikationsstrecke muss sich gut in die bestehende Software-Umgebung integrieren und wenn möglich auf nicht proprietären Technologien basieren, um eine spätere Pflege und Weiterentwicklung zu vereinfachen. Weil mit Typo3 das Quellsystem für auszuspielende Daten schon feststeht, fokussiert sich die Evaluierung auf Ausspielung, Verarbeitung und Publikation des Contents[6]. Publikationsfähige Tools, die bereits bei der Projektron GmbH eingesetzt werden, sind Adobe InDesign, Ovidius TCToolbox und Microsoft Word. In anderen Redaktionen können hierzu noch Adobe FrameMaker, Adobe InDesign Server oder andere Redaktionssysteme kommen. Die letztgenannten Systeme werden in dieser Arbeit nicht berücksichtigt.

[5] ca. 6 Kunden-Releases pro Jahr mit jeweils etwa zwei Produktbeschreibungs-Releases (Initital + Ergänzt) à zwei Varianten entsprechen 24 verschiedenen Produktbeschreibungen je Sprache in einem Jahr

[6] zur Verwendung der Bezeichnung 'Content' vgl. Drewer / Ziegler "Technische Dokumentation" (2011) : 297, in diesem Anwendungsfall also Texte, Bilder und Tabellen

4.2 Single Source Publishing

PDF und Website sollen aus dem gleichen Content-Bestand generiert werden. Diese Anforderung bringt die Schwierigkeit mit sich, dass bei einer Pflege des Contents in Typo3, alle Inhalte zunächst medienspezifisch für die Website erstellt werden. Das heißt, es findet *keine* medienneutrale Erfassung der Inhalte statt. Die Verarbeitung der Daten muss deshalb sehr flexibel sein, da sich Hierarchien und Reihenfolgen von Inhalten in der Regel zwischen Online- und Printmedien unterscheiden. Auch die Erstellung und Pflege von Website-Content verbleibt oft nicht mehr ausschließlich in der Technischen Redaktion eines Unternehmens. Deshalb darf auch eine Nachbearbeitung der aus Typo3 ausgespielten Inhalte nur begrenzt möglich sein, da die meisten Lösungen ein Zurückspielen von Änderungen in die Quelldaten nicht vorsehen und sonst Inkonsistenzen und Redundanzen auftreten können.

4.3 Strukturierte Quelldaten

Das Datenaustauschformat muss (semi-)strukturiert und aus Typo3 ausspielbar bzw. exportierbar sein. Standardformat für diesen Fall ist XML[7]. Typo3 unterstützt zwar nativ den Export nach XML, allerdings nur von einzelnen Seiten und Ordnern, die den Seitenhierarchien entsprechen. Diese Exporte sind eher für Backupzwecken oder einmalige Datenaustauschaktionen vorgesehen und müssen einzeln im Backend angestoßen werden, sind für die Extraktion von individuellen Teilbäumen also ungeeignet. Deshalb muss gerade an dieser wichtigen Stelle eine zufriedenstellende Lösung gefunden werden[8].

4.4 Automatisierung

Das Publishing von strukturierten Daten wird durch eine weitgehende Automatisierung erst wirtschaftlich. An dieser Stelle wird die Arbeitszeit, die vorher an einer Publikation aufgewendet wurde, nun eingespart. Ein manuelles Bearbeiten der Daten zwischen Ausgabe und Verarbeitung ist zeitaufwändig, fehleranfällig und mindert die Vorteile einer Vollautomatisierung erheblich. Zwar kann medienspezifisch nachgearbeitet werden (Absätze an verfügbaren Platz anpassen, Bilder für Print optimieren, etc.) doch gerade diese Möglichkeiten erweisen sich auch als problematisch. So kann es schnell zu unterschiedlichen Varianten des Contents kommen, deren Unterschiede und Änderungen später nicht mehr nachvollziehbar sind. Wird der Content ab der Eingabe in Typo3 nur noch maschinell verarbeitet kann das Single-Source-Prinzip eingehalten werden. Allerdings stellt dies auch hohe Anforderungen an Zuverlässigkeit und Genauigkeit der verarbeitenden Skripte. Konkrete Anforderung an die Publikationsstrecke ist entsprechend eine weitgehende Automatisierung unter Berücksichtigung der Datenintegrität.

---

[7] vgl. Benlian et al. (2005): "Verbreitung, Anwendungsfelder und Wirtschaftlichkeit von XML in Verlagen - Eine empirische Untersuchung"

[8] Typo3 kann auch direkt in PDF exportieren - allerdings nur einzelne Seiten ohne vorgelagerte Verarbeitungsmöglichkeiten.

4.5 Anspruchsvolles Layout

Unternehmen verfolgen mittlerweile den Ansatz, alle externen Publikationen, darunter auch Produktbeschreibungen und Dokumentation, als potentielle Marketingmaterialien zu betrachten[9]. Dieser Anspruch fordert ein ansprechendes Layout, das in der Regel mit den Vorgaben eines Corporate Design übereinstimmen muss. Neben Grafiken, müssen Tabellen und Schriften beachtet werden. Auch an Satzspiegel, Zeilenabstände und Positionierungen werden genaue Vorgaben gestellt. Die Endprodukte automatisierter Publikationsstrecken dürfen sich nur noch geringfügig von Informationsprodukten unterscheiden, die mit DTP-Programmen erzeugt wurden.

4.6 Varianten

Das Produktmanagement eines Unternehmens hat oft den Wunsch aus dem gleichen Content-Bestand, Varianten für verschiedene Zielgruppen oder Anwendungsfälle zu publizieren. Dies lässt sich auch in einer automatisierten Publishing-Umgebung realisieren. Sind die Unterschiede zwischen den Varianten nur struktureller oder gestalterischer Art, entstehen für den erstellenden Redakteur keine Mehraufwände für das Publizieren einer Variante. Bei der Projektron GmbH wurde die Produktbeschreibung bereits mit Hilfe des TCToolbox- Variantenmanagements in zwei verschiedenen Varianten publiziert: Eine ausführliche und eine kompakte Version. Diese Möglichkeit soll auch weiterhin bestehen.

4.7 Mehrsprachigkeit

International ausgerichtete Unternehmen publizieren Ihre Informationsprodukte fast immer in mehrere Sprachen. Bei der Projektron GmbH wurde bisher in vier Sprachen publiziert: Deutsch, Englisch, Französisch und Spanisch. Allerdings wurden fremdsprachige Versionen auf Grund der hohen Übersetzungskosten nur sporadisch aktualisiert. Die neue Publikationsstrecke sollte mit dem Typo3-Sprachmanagement[10] zusammenspielen und somit auch von dessen Übersetzungsmanagement-Funktionen profitieren. So können bei geringfügigen Änderungen auch nur kleine Unterknoten zum Übersetzen ausgecheckt werden, die sonst zu Modulen zusammengefasst worden wären. Die Verarbeitung des fremdsprachigen Contents soll über Parameter gesteuert werden können und sprachspezifische Besonderheiten (Silbentrennung, Sonderzeichen, etc.) beherrschen.

4.8 Verarbeitung der Daten

Die verwendete Lösung sollte die Verarbeitung komplexer Strukturen beherrschen. In Bezug auf den konkreten Anwendungsfall sind das insbesondere verschachtelte Hierarchien und Tabellen. Auch der Datenzugriff auf Typo3 ist nicht trivial: Wird über TypoScript oder direkt per SQL auf die Daten zugegriffen, müssen mehrere Tabellen miteinander verknüpft werden um Abhängigkeiten zu erkennen und Inhalte zu aggregieren. Auch die Art des Feldinhalts innerhalb der von Typo3 verwalteten Tabellen unterscheidet sich stark. Manche Erweiterungen speichern reinen Text,

[9] www.tekom.de/artikel/artikel_556.html

[10] http://wiki.typo3.org/De:Multi_language_sites

andere HTML. Typo3 selbst verwendet ein XML-ähnliches-Format. Bei der Extraktion des Contents muss deshalb auf Flexibilität bei gleichzeitiger Datenintegrität geachtet werden.

4.9 Rechtliche Verbindlichkeit

Gerade Dokumentationen und Produktbeschreibungen dienen als rechtliche Grundlage beim Kauf eines Produktes. Sie beschreiben den Funktionsumfang und dienen als verbindliche Dokumente. Deshalb muss sichergestellt werden, dass gerade bei einer automatisierten Publikation keine Fehler vom System verursacht werden (falscher Tabellenaufbau, abgeschnittener Text, vertauschte Bilder, etc.).

5 Möglichkeiten: Datenausgabe

5.1 Extension von Dienstleister

Da die Zusammenarbeit mit ID.on bereits beim Relaunch der Website erfolgreich war, die Agentur eigene Typo3-Entwickler beschäftigt und den Aufbau der Projektron-Installation kennt, wurde eine Anfrage mit der Problemstellung zum XML-Export aus Typo3 gestellt. Angeboten wurde eine Reihe von Lösungsansätzen, welche die Entwicklung einer Extension zum XML-Export vorsahen. Aufgrund der hohen Aufwände (bis zu 20 Programmierer-Tage), die eine solche Entwicklung gekostet hätte, wurde diese Möglichkeit wieder verworfen.

5.2 Lösungen von Drittanbieter

Der Markt von Drittanbieter-Erweiterungen im Typo3-Umfeld ist groß. Allerdings konzentrieren sich fast alle Lösungen auf die Ausgabe eines Endformates, nicht auf das Ausspielen eines Zwischenformates wie XML. So gibt es diverse Extensions zum direkten App-Publishing oder auch zum Publizieren in PDF[11]. Allerdings begrenzen sich hier die Möglichkeiten oft nur auf einzelne Seiten, nicht auf selektive Seitenbäume, wie es in diesem Anwendungsfall benötigt wird.

Anbieter, die sich auf den Web-to-Print-Bereich spezialisiert haben bieten zwar Lösungen an, die den Anforderungen entsprechen[12], doch diese individualisierten Entwicklungen (die fast immer auch die Stylesheet-Entwicklung beinhalten) sind sehr teuer und für diesen Anwendungsfall nicht wirtschaftlich genug.

5.3 Eigenentwicklung TypoScript-Template

Wie schon eingangs erwähnt ist es möglich mit Hilfe der Typo3-eigenen Konfigurationssprache TypoScript eigene Templates für die Seitenausgabe zu scripten. Diese Templates können mit dem

[11] http://www.typo3-macher.de/typo3-extension-pdf.html

[12] http://www.cross-media.net/de/

Setzen entsprechender Header und Strukturierung des Inhalts auch XML-Seiten ausgeben. Da dies aber nicht der vorgesehenen Funktion von TypoScript entspricht, müssen die XML-Seiten 'von Hand' aufgebaut werden. Das bedeutet, im Template werden Tags definiert, die mit Inhalten der Datenbank gefüllt werden. Nachteil dieser Lösung ist der große Scripting-Aufwand und das nötige Erlernen von TypoScript und den Typo3-spezifischen Datenbankstrukturen. Vorteil ist allerdings die extrem hohe Flexibilität in der Ausgabe.

6 Möglichkeiten: Datenverarbeitung

6.1 InDesign

In der Redaktion wurde bereits eine halbautomatisierte Publikationsstrecke von Anwenderberichten in InDesign realisiert. Diese Lösung basiert auf InDesign-Templates, die mit Daten aus einer CSV-Datei befüllt werden. Ebenso ist es möglich, mit InDesign XML-Dateien einzulesen und diese mit einem internen Scripting-Framework auf Basis von JavaScript umzustrukturieren und zu verarbeiten[13]. Diese Lösung wurde zu Beginn der Evaluierungsphase bevorzugt, da man so auf schon vorhandene InDesign-Templates zurückgreifen konnte und eine hohe gestalterische Freiheit bei gleichzeitiger Bedienfreundlichkeit hat. Gegen diese Lösung spricht allerdings, dass so nur ein halbautomatisierte Publishing realisiert werden könnte und das interne Scripting-Konzept von InDesign erlernt werden müsste.

6.2 JODReports

Eine weitere Möglichkeit die Daten aus Typo3 zu verarbeiten stellt das kostenlose Framework JODReports[14] dar, das XML-Daten auf Basis von OpenOffice-Templates verarbeiten kann. Basierend auf Java können so Daten vorher verarbeitet werden und anschließend in einem Template-basierten Ansatz in PDF publiziert werden. Vorteile dieses Konzepts stellt das einfache Templating dar, Nachteile allerdings das bisher fehlende Know-How in Java und die Notwendigkeit, neben dem schon eingesetzten Microsoft Word, Open Office als weiteres Textverarbeitungsprogramm nur für das Templating betreiben zu müssen.

6.3 XSLT / XSL-FO

Die in der Technischen Dokumentation bevorzugte Methode XML-Dateien als PDF-Dokumente zu publizieren ist XSL-FO, eine Seitenfluss-Beschreibungssprache. Zusammen mit XSLT bietet diese Technologie einen mächtigen Funktionsumfang, was Verarbeitung, Umstrukturierung und Manipulation von Daten angeht[15]. Auch gestalterisch können, mit entsprechenden Kenntnissen von XSL-FO, die Anforderungen an die zu publizierende Produktbeschreibung erfüllt werden. Nachteile dieser Technologie sind die hohen Aufwände bei der Erstellung von Stylesheets und die Tatsache,

[13] http://help.adobe.com/de_DE/indesign/cs/using/WS372C59DB-BD13-4806-A399-794E754FF37Aa.html

[14] http://jodreports.sourceforge.net/

[15] http://www.w3.org/Style/XSL/

dass Änderungen nur mit guten Kenntnissen der Sprache vorgenommen werden können. Vorteile sind Flexibilität, hohe Automatisierungsfähigkeit und der Einsatz einer standardisierten Technologie.

7 Lösung

7.1 Ausgabe: TypoScript-Template

Für die Ausgabe von XML aus Typo3 wurde eine eigene Lösung auf Basis einen TypoScript-Templates entwickelt. Diese Idee ist nicht neu[16], jedoch in diesem Umfang und unter diesen Voraussetzungen noch nicht (auffindbar) dokumentiert worden. Bei TypoScript- Templates handelt es sich im Wesentlichen um Konfigurationen, die in ein PHP-Array umgewandelt und vom Typo3-Core-Framework verarbeitet werden. Dabei werden zunächst Objekte definiert[17], welchen anschließend Datenbankzugriffe zugeordnet werden. Zusätzlich bietet die Sprache weitere typische Möglichkeiten einer klassischen Scripting-Sprache wie Variablen, Bedingungen oder Schleifen. Im Folgenden will ich grundlegende Arbeitsweise bei der Template-Entwicklung darstellen:

7.1.1 Grundgerüst

```
xmlseite = PAGE

xmlseite {
  typeNum=1312
  config.disableAllHeaderCode = 1
  config.metaCharset = utf-8
  config.additionalHeaders = Content-Type:text/xml;charset=utf-8
  config.xhtml_cleaning = 0
  config.admPanel = 0

  1 = TEXT
  1.value = <?xml version="1.0" encoding="utf-8" standalone="yes"?>

 # weiteres Script....
```

Zunächst wird ein neuer Seitentyp (`PAGE`) und seine zugehörige Nummer (`1312`)[18] definiert, welcher später über den URL-Parameter `?type=1312` aufgerufen werden kann. Anschließend wird ein XML-Header festgelegt und dem erstes Objekt (`1`) der Typ `TEXT` mit Inhalt eines XML-Prologs zugewiesen. Bei der weiteren Erstellung von Objekten muss nun darauf geachtet werden, wohlgeformtes XML zu erzeugen.

[16] http://t3n.de/magazin/valides-xml-typoscript-erzeugen-bereitstellung-221171/

[17] In TypoScript werden Objekte als Zahlen entsprechend ihrer Reihenfolge in der Ausgabe repräsentiert.

[18] Beliebige ganze Zahl mit Ausnahme von `0` (Standard-HTML) und `98` (Druckausgabe).

7.1.2 Content-Abfragen

Um Inhalte über den speziellen Seitenbaum im Funktionsbereich der Website[19] zu aggregieren musste ein eigenes Informationsmodell erdacht werden, das im Template aufgebaut wird.

```
2 = CONTENT
2 {
  table = pages
  select {
      pidInList = 604
      orderBy = sorting
      andWhere = uid != 615
      }
  renderObj = COA
  renderObj {

# Objekt das gerendert werden soll....
```

In der Abbildung ist eine typische Datenbankabfrage auf höchster Ebene zu erkennen. In dieser Abfrage werden in der Tabelle `pages` alle Unterseiten der Stammseite mit der `pid`[20] `604` abgefragt mit Ausnahme der Unterseite mit der `uid` `615`. Für jedes Ergebnis wird nun die `uid` der Unterseite an die `renderObj` Funktion übergeben.

```
renderObj{
  wrap = <item>|</item>
  50 = TEXT
  50{
    wrap = <title>|</title>
    field = header
    }
  60 = TEXT
  60{
    wrap = <content>|</content>
    field = bodytext
    HTMLparser{
        allowTags = h3, link
        tags.h3.remap = subtitle
        tags.link.remap = reference
        noAttrib = link
        }
```

Hier ist eine typische Methode zur Objekterzeugung dargestellt. Es wurde die `uid` einer Seite an die `renderObj` Funktion übergeben. Diese kann nun einzelne Tabellenspalten für diese `uid` auslesen. In unserem Fall werden zwei Objekte (`50` & `60`) erzeugt, denen jeweils der Wert der Tabellenspalte `header` bzw. `bodytext` zugewiesen wird. Mit `wrap` kann dieser Wert mit Tags umklammert werden, um einer XML-Syntax zu entsprechen. Ist in den Tabellenfeldern HTML oder auch XML enthalten,

---

[19] http://www.projektron.de/bcs/funktionen

[20] In Typo3 besitzen alle Seiten eine unique id (`uid`) zur eindeutigen Identifizierung und eine parent id (`pid`) zur hierarchischen Einordnung im Seitenbaum. Die `pid` entspricht der `uid` der übergeordneten Seite.

kann dieses durch eine `HTMLparser`-Funktion manipuliert werden (Entfernen und Remappen von Tags, Entfernen von Attributen, etc.).

7.1.3 Mehrsprachigkeit

Templates können beim Aufruf der Seite URL-Parameter übergeben werden, mit denen das Template gesteuert werden kann. Damit kann beispielsweise die Sprache der Quelldaten festgelegt werden. Ein Aufruf des englischen Contents hätte so die URL-Parameter `?type=1312&lang=en`, welche im TypoScript anschließend aufgegriffen werden und entsprechende globale Konfigurationsvariablen (`config.*`)gesetzt werden. Diese config-Variablen[21] steuern das Typo3-eigene Sprachmanagement.

```
[globalVar = GP:lang = de]
xmlseite {
  config.sys_language_uid = 0
  config.language = de
  config.locale_all = de_DE
 }
[global]

[globalVar = GP:lang = en]
xmlseite {
  config.sys_language_uid = 2
  config.language = en
  config.locale_all = en_EN
 }
[global]
```

7.1.4 Vorteile von TypoScript

TypoScript zum Auslesen von Datenbankinhalten zu verwenden erscheint zunächst umständlich, jedoch können einige Typo3-spezifischen Mechanismen ausgenutzt werden:

- Rechte und Sichtbarkeiten von Inhalten werden berücksichtigt
- Sprachverwaltung der Einträge durch Parameterübergaben
- Bedingte Ausgaben und Inhaltsmanipulation (Ersetzungen, Aufsplittung, etc.)

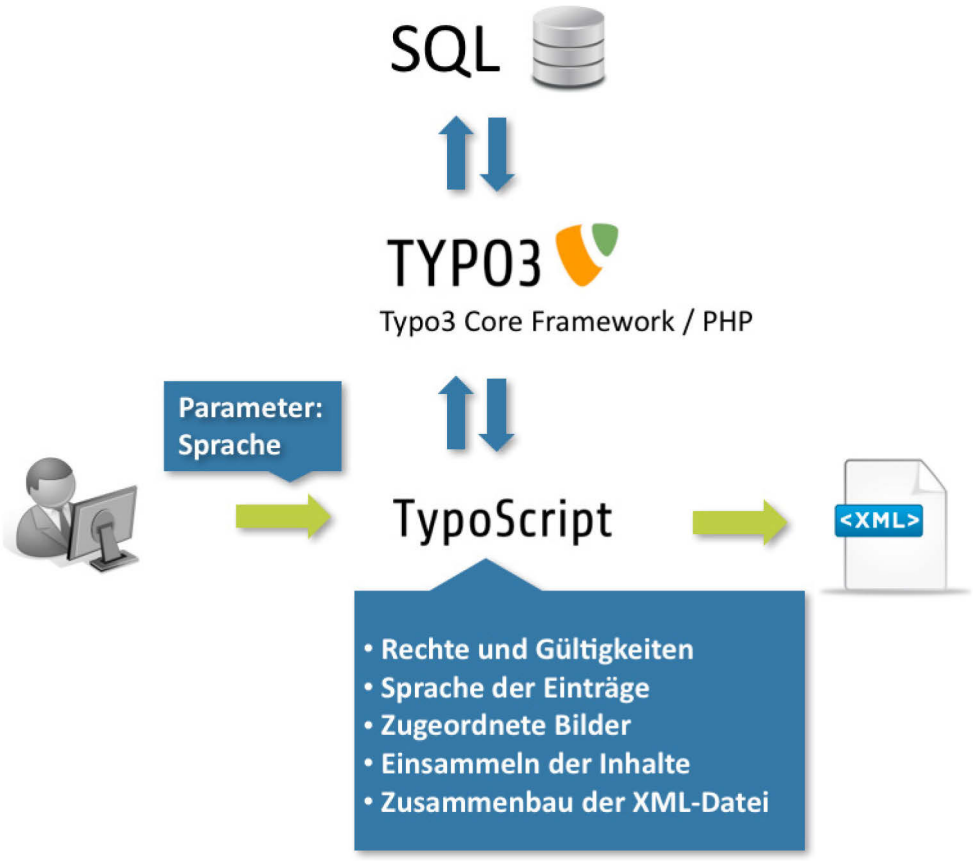
Diese tiefe Integration und die resultierenden Vorteile überwiegen die Nachteile, die durch die gewöhnungsbedürftige Syntax[22] und das komplizierte Debugging[23] hervorgerufen werden.

---

[21] Im Speziellen die `sys_language_uid`, die im Typo3-Backend bei Datenbankzugriffen die Auswahl der korrekten Tabellen für die jeweiligen Sprachen beeinflusst.

[22] Die offizielle TypoScript-Dokumentation selbst schreibt z.B.: “The "if"-function is a very odd way of returning true or false! Beware!” (Quelle: http://wiki.typo3.org/TSref/if)

[23] In diesem Anwendungsfall äußern sich Fehler im TypoScript in einem nicht validen Aufbau der XML-Ausgabe. Die invalide Stelle des XML ist oft einziger Anhaltspunkt für den Fehler.



Aufbau der Content-Ausgabe: Anfrage mit Parameterübergabe, TypoScript-Template als Gerüst für die Ausgabe und das Typo3 Core Framework als steuernder Layer für die Datenbankzugriffe

7.1.5 Bedingte Ausgaben

Bei der Einsammlung von Content über den Seitenbaum muss das Template bei Varianten flexibel reagieren können. So müssen beispielsweise bei fremdsprachigen Zugriffen auf das DAM die Datenbankabfragen in anderen Hierarchiestufen durchgeführt werden als bei deutschen Einträgen. Dafür gibt es in TypoScript die Möglichkeit Weichen durch `if`-Abfragen zu stellen, die dann zum Beispiel den mitgegeben Sprachparameter vergleichen können.

```
renderObj = COA
renderObj{
    50 = TEXT
    50{
    if{
        value.data = GPvar:lang
        equals = de
        }
    dataWrap = <image src='{field:file_path}{field:file_name}'>{field:description}</image>
        }
    }
```

7.1.6 Fertiges Template

Das fertige TypoScript-Template umfasst ca. 1000 Zeilen Code und ist angepasst auf die Projektron-spezifische Typo3-Installation. Aufgrund der speziellen Anforderungen an die Struktur, den Zugriff auf erweiterte Systembereiche und den expliziten Aufruf mancher Knotenpunkte ist eine einfache Portierbarkeit des Templates auf andere Typo3-Installationen leider nicht möglich. Eine universelle und frei verfügbare Lösung für den selektiven XML-Export aus Typo3 existiert somit auch weiterhin nicht. Allerdings hat man mit TypoScript ein mächtiges Werkzeug zur Verfügung, das schnell Erlernbar und auf die eigenen Bedürfnisse anpassbar ist.

7.1.7 Erzeugte XML-Datei

Wurden im TypoScript alle Objekte richtig gerendert und eine wohlgeformte Struktur eingehalten, wird beim Aufruf der Seite mit den entsprechenden URL-Parametern, valides XML ausgegeben. Die Ausgabe enthält in unserem Anwendungsfall (mit Ausnahme von printspezifischen Inhalten) alle relevanten Inhalte, die für die Generierung der Produktbeschreibung nötig sind. Der Umfang beträgt über 2000 Zeilen Ausgabe pro Sprache mit jeweils 70 zugeordneten Bildern.

```
<root>
    <section class="Funktionen">
        <function>
            <detail version="7.0">
                <title>Projektportfolio-Auswertungen</title>
                <content>
                    <subtitle>Für strategische Entscheidungen</subtitle>
                    <block>Welche Projekte sind für Ihr Unternehmen besonders wichtig? Projektron BCS vergleicht die
                    Projekte anhand ihrer Komplexität sowie deren Nutzen für Ihr Unternehmen. Die grafische Auswertung
                    in einer Matrix veranschaulicht den unternehmerischen Wert Ihrer Projekte und unterstützt Sie bei
                    strategischen Entscheidungen bezüglich Ihres Projektportfolios.</block>
                </content>
                <image src="/example/portfolio_auswertung.png">Portfolio-Auswertung: Für Ihre Strategie</image>
            </detail>
```

7.2 Verarbeitung: XSLT/XSL-FO

Zur Verarbeitung der generierten XML-Daten kommt ein XSL-Stylesheet zum Einsatz. In dieser Datei sind Angaben zur Transformation der Daten durch einen XSL-Transformator enthalten, der das FO-Zwischenformat erzeugt. Dieses wird wiederum von einem FO-Prozessor zu einem PDF verarbeitet. In der Regel werden beide Schritte innerhalb des FO-Prozessors ausgeführt.

7.2.1 Umgebung

Neben dem kostenpflichtigen Marktführer AntennaHouse Formatter steht als kostenlose Alternative der Open Source FO-Prozessor Apache FOP[24] zur Verfügung, welcher zwar nur einen Teil des XSL-FO-Standards beherrscht, für die Anforderungen der Projektron GmbH aber völlig ausreicht. Als XSL-Transformator kommt Apache Xalan[25] zum Einsatz, der im FOP-Paket bereits integriert ist. Zur Registrierung von alternativen Fonts, müssen sogenannte Font-Metrics-Dateien für jeden Schriftschnitt des Fonts erstellt werden. Diese können zum Beispiel mit dem Java-

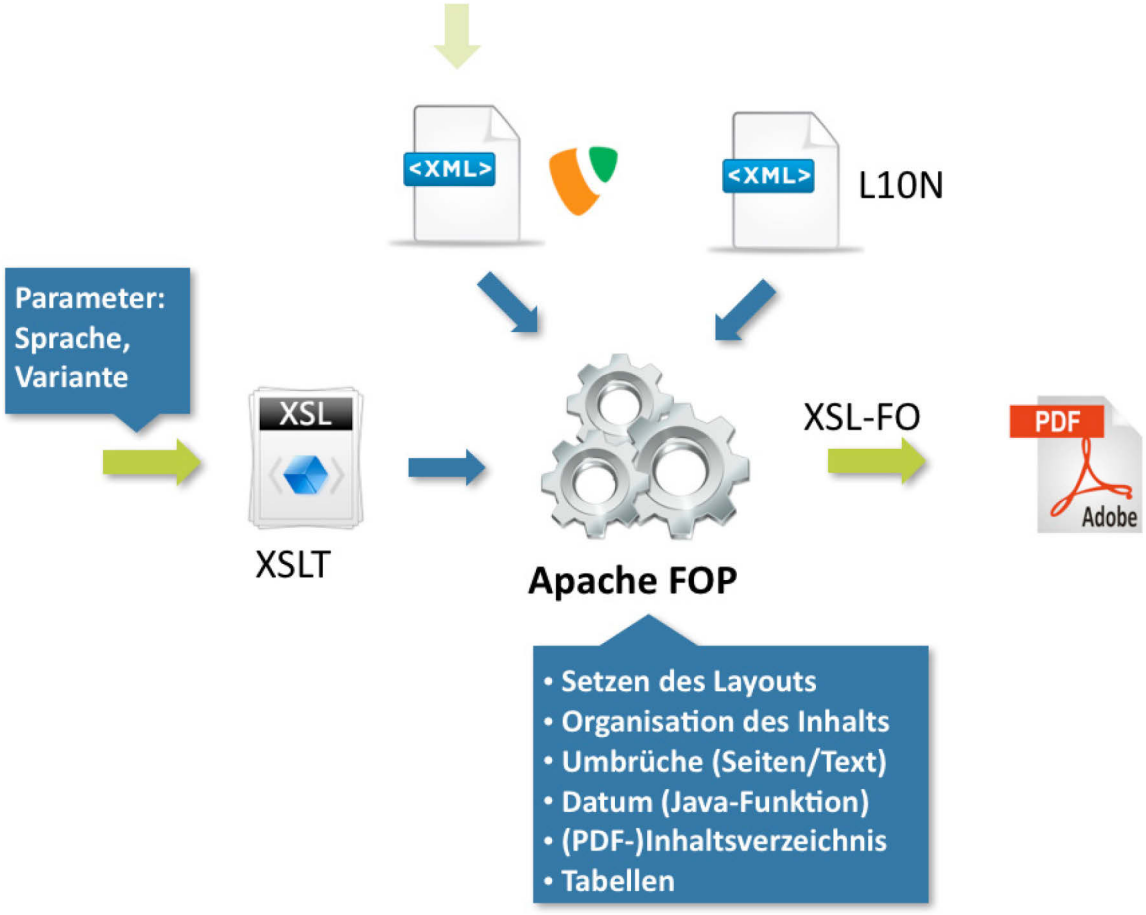
[24] http://xmlgraphics.apache.org/fop/

[25] http://xml.apache.org/xalan-j/ (in FOP enthalten)

Programm TTFReader[26] aus TrueType-Fonts generiert werden. Für die sprachspezifische Silbentrennung werden sogenannte Hyphenation-Patterns benötigt, die vom Open-Source-Projekt offo[27] zur Verfügung gestellt werden und in FOP eingebunden werden können.

7.2.2 Aufgaben

Neben dem Umsortieren von Einträgen und Hinzufügen von Inhalten aus ausgelagerten, statischen XML-Dateien können auch ein Inhaltsverzeichnis und Tabellen erstellt oder ein Griffregister generiert werden. Über eine eingebundene Java-Funktion kann für organisatorische Zwecke das aktuelle Datum hinzugefügt werden. Um auf die unterschiedlichen Lauflängen der Sprachen zu reagieren, können Bedingungen die Textlängen innerhalb begrenzter Blöcke prüfen und gegebenenfalls die Schriftgröße anpassen. Auch das Einfügen von Querverweisen oder das Hervorheben neuer Einträge kann bei der Verarbeitung erledigt werden.



Verarbeitung der XML-Quelldatei zu einem PDF durch Apache FOP / Apache Xalan

[26] http://xmlgraphics.apache.org/fop/0.95/fonts.html#truetype-metrics (in FOP enthalten)

[27] http://offo.sourceforge.net/hyphenation/index.html

7.2.3 Varianten

Über Parameter kann eine Variantenerzeugung gesteuert werden. So kann das Stylesheet beim Aufruf verschiedene Ausgabevarianten des Contents erzeugen oder der Zielsprache angepasst werden. Diese Parameter werden dem XSL-Transformator beim Aufruf mitgegeben und stehen im Stylesheet als mögliche Vergleichswerte oder Variablen zur Verfügung. Bei der Transformation der Projektron-Produktbeschreibung werden die Parameter `Typ` und `Sprache` berücksichtigt. Der Parameter `Typ` hat hierbei Auswirkungen auf den strukturellen Aufbau der Produktbeschreibung, d.h. Anordnung und hierarchische Tiefe der Inhalte. Der Parameter `Sprache` auf die sprachabhängigen statischen Grafiken.

Vergleich zwischen strukturellen Varianten: Gleiche Content-Elemente sind grün hinterlegt

7.2.4 Mehrsprachigkeit

Das Stylesheet kann entsprechend Ausgabe von Typo3 auf die verschiedenen Sprachen der Quelle reagieren. Gesteuert wird das über den Parameter `Sprache`, der zum Beispiel bewirkt, dass andere Regeln zur Silbentrennung angewandt werden.[28] Auch können noch nicht übersetzte Bereiche der Website für eine Sprache ausgeblendet werden oder durch englischen Content ersetzt werden. Auch statischer Content kann entsprechend der Zielsprache aus externen Dateien eingefügt werden.

[28] Die Auswahl der Hyphenation-Patterns für die jeweilige Zielsprache wird über das `xml:lang`-Attribut an `fo:block`-Elementen gesteuert. Das Attribut wird mit dem Parameter `Sprache` verknüpft.

7.2.5 Statische Informationen

Nahezu der komplette Content für die Produktbeschreibung kommt aus Typo3. Manche Inhalte, die nur für die Printausgabe benötigt werden, sind dort aber nicht verfügbar. Sie müssen zusätzlich angegeben werden. Beispiele sind Pfadangaben zu Deckblatt- oder Kapitelgrafiken, spezielle Aufzählungssymbole oder auch statische Strings, die im Impressum verwendet werden. Diese ausgelagerten Angaben müssen in vier Sprachen für zwei Varianten vorliegen. Es ist möglich diese Informationen innerhalb des Stylesheets zu pflegen, allerdings erschwert dies den Wartungsaufwand und die Komplexität, da schon ab zwei Sprachen Bedingungen eingeführt werden müssen, um sprachspezifische Inhalte unterscheiden zu können. Deswegen empfiehlt sich die Auslagerung in eine lokale XML-Datei. Dadurch wird auch der Export von noch nicht übersetzten Strings an einen Übersetzungsdienstleister vereinfacht.

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!DOCTYPE l12n SYSTEM "l12n.dtd">
<l12n>
    <labelset lang="all">
        <label name="Firma" type="Alle">Projektron GmbH</label>
        <label name="Strasse" type="Alle">Charlottenstra&#x00df;e</label>
        <label name="Hausnr" type="Alle">68</label>
        <label name="PLZ" type="Alle">10117</label>
        <label name="Ort" type="Alle">Berlin</label>
        <label name="Telefon" type="Alle">+49 30 3 47 47 64-0</label>
    </labelset>
    <labelset lang="de">
    </labelset>
    <labelset lang="en">
    </labelset>
    <labelset lang="fr">
        <label name="Impressum" type="Alle">Mentions légales</label>
        <label name="Herausgeber" type="Alle">Gérant</label>
        <label name="URL_kurz" type="Alle">www.projektron.fr</label>
        <label name="URL_lang" type="Alle">http://www.projektron.fr</label>
        <label name="Info_Mail" type="Alle">info@projektron.de</label>
        <label name="Stand" type="Alle">Mise à jour</label>
        <label name="Rechtliche_Hinweise" type="Alle">Avis juridique</label>
```

Auszüge aus der XML-Datei mit den ausgelagerten statischen Informationen.

7.2.6 Datenquellen

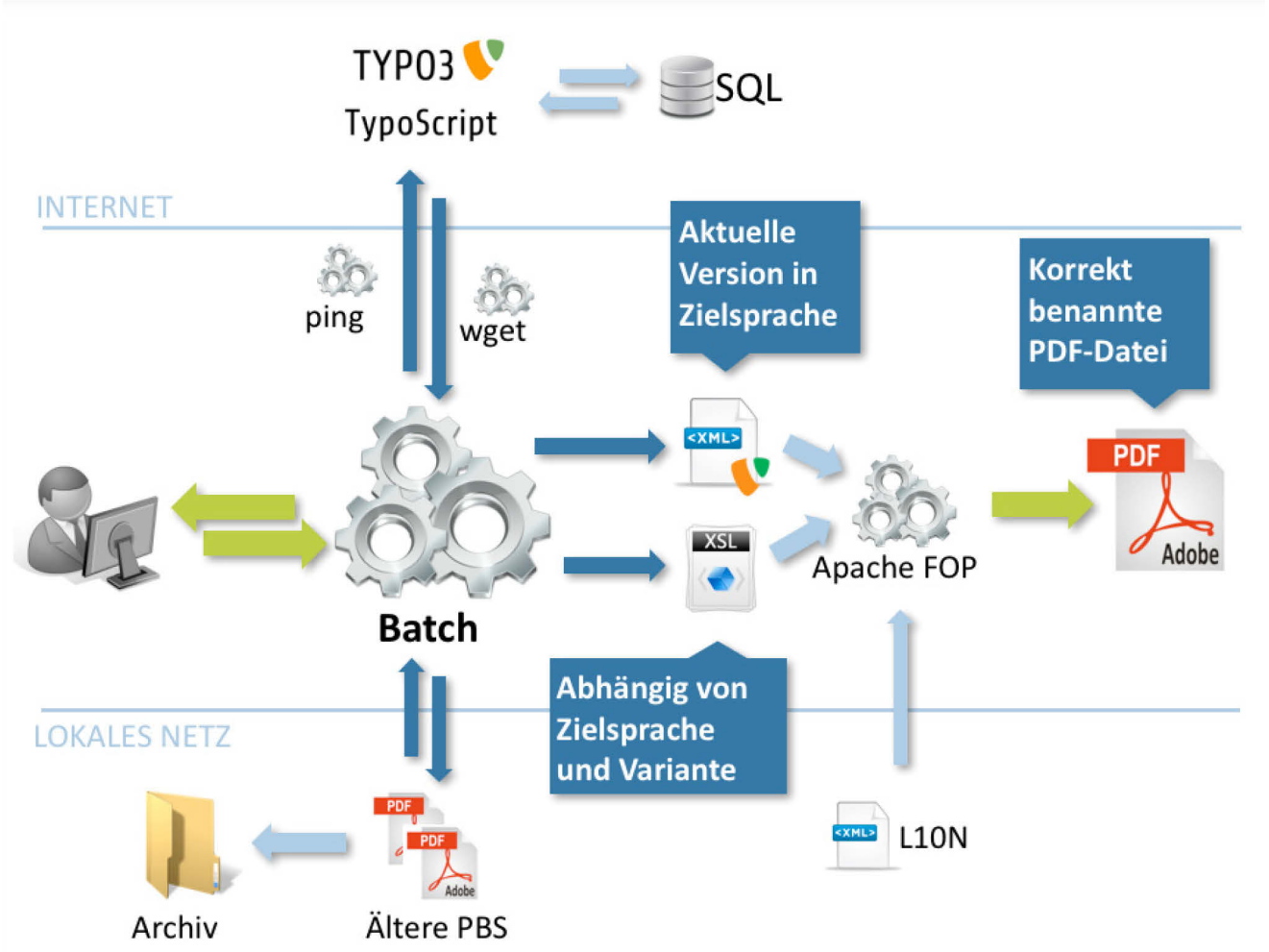
Aus den oben genannten Gründen wird neben Typo3 noch eine zusätzliche Datenquelle benötigt. Für die Produktbeschreibung wurde hierfür eine XML-Datei angelegt, die alle benötigten Angaben, unterteil nach Zielsprachen und Publikationsvarianten, enthält. Der Zugriff auf die Datei erfolgt zu Beginn des Stylesheets über die `document()`-Anweisung und einem, mit den Übergabeparametern gebildeten XPath-Ausdruck. Die Ergebnisse werden anschließend in globale Variablen geschrieben, die im gesamten Stylesheet angewendet werden können.

```
4   <!-- Übergebene Parameter -->
5       <xsl:param name="Typ">Lang</xsl:param>
6       <xsl:param name="Sprache">de</xsl:param>
7
8   <!-- Ausgelagerte Strings abhängig von Parameter Sprache und Typ -->
9       <xsl:variable name="Header" select="document('l12n.xml')
        //labelset[@lang=$Sprache]/label[@type=$Typ][@name='Header']"/>
10      <xsl:variable name="Impressum" select="document('l12n.xml')
        //labelset[@lang=$Sprache]/label[@nam ='Impressum']"/>
11      <xsl:variable name="Herausgeber" select="document('l12n.xml')
        //labelset[@lang=$Sprache]/label[@name='Herausgeber']"/>
```

Zugriff auf ausgelagerte Informationen über `document()`

7.3 Automatisierung

Bis jetzt sind XML-Export aus Typo3 und XML-Verarbeitung durch FOP noch zwei getrennte Prozesse. Um eine vollständige Automatisierung zu erreichen, müssen die beiden Arbeitsschritte miteinander verknüpft werden.



Eine Batch-Datei dient als Steuerungszentrale für den Publikationsprozess

7.3.1 Batch

Eine Batch[29]-Datei dient als Benutzerschnittstelle und Steuerungszentrale des gesamten Publikationsprozesses. Beim Aufruf werden die benötigen Parameter vom Benutzer abgefragt. Bei der Produktbeschreibung sind das `Typ` und `Sprache`. Anschließend können eine Reihe von Aktionen abgehandelt werden, zum Beispiel:

- Prüfen ob Typo3 erreichbar ist und eine Internetverbindung besteht
- Ältere Publikationen erkennen und archivieren
- XML-Quelle in der entsprechenden Sprache herunterladen
- Apache FOP mit entsprechenden Parametern aufrufen
- Fertige PDF benennen mit Variante, Sprache und Publikationsdatum
- Ergebnis nach fertiger Publikation öffnen

Natürlich ist es ebenso möglich mit einer Batch-Datei, alle Sprachen und Varianten auf einmal zu publizieren. Dann sind keine Benutzerangaben nötig und die Aktualisierung der Publikation kann beispielsweise als Cronjob ausgeführt werden.

7.3.2 Hilfsprogramme

Neben den zwei Haupt-Applikationen Typo3 und Apache FOP sowie der Microsoft Batch arbeiten noch zwei kleine Programme im Hintergrund:

7.3.2.1 ping

Das Werkzeug ping[30] wird verwendet um zu prüfen, ob der Host, auf dem Typo3 installiert ist, erreichbar ist. Ist dies nicht der Fall, kann durch ein entsprechendes Error-Handling die angestoßene Publikation frühzeitig abgebrochen und dem Nutzer eine Rückmeldung gegeben werden. Ping kann auch entgegen seiner eigentlichen Funktion dafür eingesetzt werden, kurze Verzögerungen einzubauen, um zum Beispiel Zwischenmeldungen anzuzeigen.

7.3.2.2 Wget

Um die XML-Ausgabe von Typo3 herunterzuladen und in eine Datei zu schreiben wird das Kommandozeilen-Programm Wget[31] verwendet. Bei Wget handelt es sich um freie Software aus dem GNU-Projekt, die für nahezu jedes Betriebssystem verfügbar ist. Alternativ kann auch cURL[32] eingesetzt werden, welches zusätzlich noch eine Upload-Funktion zur Verfügung stellt. Damit könnten Publikationen sofort nach der Fertigstellung im Internet zur Verfügung gestellt werden

[29] Im konkreten Beispiel handelt es sich um eine Windows Batch-Datei, es können allerdings auch andere Arten der Stapelverarbeitung zum Einsatz kommen.

[30] http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/ping.mspx

[31] http://www.gnu.org/software/wget/

[32] http://curl.haxx.se/

```
IF "%lang%"=="" set lang=%1
IF "%lang%"=="" set /p lang= Welche Sprache (de, en, fr, es)?
echo.
IF "%type%"=="" set type=%2
IF "%type%"=="" set /p type= Welche Variante (lang, kurz)?
IF "%type%"=="lang" set publication=Produktbeschreibung
IF "%type%"=="kurz" set publication=Kurzbeschreibung

SET HOSTNAME=www.projektron.de
SET PDFNAME=../ProjektronBCS_%publication%_%lang%_%DATE%.pdf

ping %HOSTNAME%
if "%errorlevel%"=="0" (
        wget -O produktbeschreibung_%lang%.xml "http://%HOSTNAME%/?type=1312&lang=%lang%"
        fop -q -dpi 300 -c conf/fop.xconf -r -xml produktbeschreibung_%lang%.xml -xsl
            produktbeschreibung.xsl -param Sprache %lang% -param Typ %type% -pdf %PDFNAME%
        start "pdf" %PDFNAME%

        ...
```

Auszüge aus der steuernden Batch-Datei: ping, Wget und FOP arbeiten zusammen.

7.3.3 Stabilität

Die Verarbeitung durch das Stylesheet läuft sehr stabil. Probleme bei der Verarbeitung durch die Templates im Stylesheet kommen nicht vor, da die Quelldaten selbst regelbasiert aufgebaut werden. Der Zugriff auf die Typo3-Datenbank ist in der Regel auch unproblematisch, da verwaltende Tätigkeiten wie Deaktivieren und Entfernen von Einträgen sowie die Gültigkeiten und Zugriffsrechte vom Typo3-Core-Framework übernommen werden. Updates von Typo3 sollten aber grundsätzlich immer zuerst auf einer Testinstanz installiert und getestet werden. Wird die interne Ablagestruktur umstrukturiert oder statische PIDs verändert, kann es zu Problemen beim Zugriff durch das TypoScript-Template kommen.

8 Fazit

Die vorgestellte Lösung zeigt, dass es möglich ist mit freier Software und verhältnismäßigem Aufwand, eine voll automatisierte Publikationsstrecke aufzubauen, die alle geforderten Anforderungen erfüllt. Der durchgängige Einsatz von OpenSource-Technologien bringt sowohl Vor- als auch Nachteile mit sich. Nachteile sind der eingeschränkte Funktionsumfang von Apache FOP (im Gegensatz zu kommerziellen Formattern) und die teilweise sehr umständliche Einrichtung und Konfiguration. Entscheidende Vorteile sind allerdings das Wegfallen von Anschaffungskosten und die Unabhängigkeit von proprietären Formaten und Erweiterungen.

Die Umsetzung wurde oft nach dem Trial-and-Error-Prinzip durchgeführt, da es zu einer solchen Lösung nahezu keine veröffentlichten Informationen verfügbar sind. Allerdings hat sich der initiale Aufwand gelohnt. Die Publikationsstrecke läuft bei Projektron seit mittlerweile fast einem Jahr und wird ständig um Funktionen erweitert.

Eidesstattliche Erklärung

Ich erkläre ehrenwörtlich,

1. dass ich die vorliegende Arbeit selbstständig verfasst habe;
2. dass ich die Übernahme wörtlicher Zitate aus der Literatur sowie die Verwendung der Gedanken anderer Autoren an den entsprechenden Stellen innerhalb der Arbeit gekennzeichnet habe;
3. dass ich meine Arbeit bei keiner anderen Prüfung vorgelegt habe.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Jan Oevermann
Karlsruhe, 10.12.2012