Darstellungen von TEI-Dokumenten für E-Reader im EPUB-Standard

Diplomarbeit

im Diplomstudiengang Informatik am Fachbereich 16 Elektrotechnik/Informatik der Universität Kassel

Vorgelegt am 15. Februar 2011 von

Mohammed Abuhaish

Erstgutachter: Prof. Dr. Lutz Wegner

Zweitgutachter: Prof. Dr. Albert Zündorf

Erklärung

Ich versichere hiermit, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Kassel, den 15.02.2011 Mohammed Abuhaish

Vorwort

Die vorliegende Diplomarbeit entstand im Zeitraum von November 2010 bis Februar 2011 und befasst sich mit der Konzeption eines Moduls, das die Darstellung von TEI-Dokumenten im EPUB-Standard für E-Reader ermöglicht. Dieses Thema ergab sich aus einem interdisziplinären Projekt zwischen dem Fachbereich 2 und dem Fachbereich 16 der Universität Kassel. Daher möchte ich zunächst Herrn Dr. des. Christof Schöch danken, der das Projekt initiiert hat.

Besonderer Dank gilt Herrn Prof. Dr. Lutz Wegner und Herrn Dipl.-Inform. Dipl.-Math. Sebastian Pape, die mir für das Überprüfen der Arbeitsergebnisse einen E-Reader zur Verfügung stellten und diese Diplomarbeit mit großem Engagement betreut haben.

Schließlich möchte ich auch Herrn Prof. Dr. Albert Zündorf danken, der sich bereiterklärte, das Zweitgutachten für diese Arbeit zu erstellen.

Kassel, Februar 2011 Mohammed Abuhaish

Inhaltsverzeichnis

1 Einleitung

1.1 Vorarbeit

Die Grundlage dieser Diplomarbeit bildet das Projekt „Visualisierung mit XSL für den Bereich Textedition“, geleitet von Dr. des. Christof Schöch, wissenschaftlicher Mitarbeiter in französischer Literaturwissenschaft am Institut für Romanistik (Fachbereich 2) der Universität Kassel. Ziel dieses Projekts war es, eine frei zugängliche und kommentierte elektronische Ausgabe des im Jahr 1776 erschienenen Werks „Essai sur le récit, ou Entretiens sur la manière de raconter“ von François-Joseph Bérardier de Bataut zu verfassen [1]. Die in Dialogform geschriebene Abhandlung widmet sich der Kunst des Erzählens „in fiktionalen und nicht-fiktionalen Texten im französischen 18. Jahrhundert“ und besitzt „für die Geschichte der Reflexion über das Erzählen und für die Geschichte des Romans“ einen besonderen Stellenwert [1].

Im Wintersemester 2009/10 begann das interdisziplinäre Projekt, das von Prof. Dr. Lutz Wegner und Dipl.-Inform. Dipl.-Math. Sebastian Pape vom Fachbereich 16 Praktische Informatik betreut wurde. Die Studenten der Informatik Dmitrij Funkner, Roman Kominek und der Autor der vorliegenden Arbeit, Mohammed Abuhaish, entwickelten für die elektronische Edition des oben genannten Werkes, vorliegend im XML/TEI-Format, ein XSL-Stylesheet, das die Visualisierung der TEI-Dokumente ermöglicht. Bei der Konzeption des Stylesheets war es wichtig, die verschiedenen Arten der Textdarstellung zu berücksichtigen, damit die Charakteristika eines Textes erhalten bleiben: Handelte es sich z.B. um ein Gedicht, sollte es für jede Zeile einen Zeilenumbruch geben und eine kleinere Schriftgröße gewählt werden. Zudem wurde die Umsetzung dynamischer Inhalte mit Javascript realisiert, Beispiele hierfür sind das Umschalten zwischen der Originalfassung und der korrigierten Version, das Aufklappen von Randbemerkungen (Anmerkungen des Autors bzw. des Herausgebers) sowie das direkte Aufrufen einer bestimmten Seite.[1] Die TEI-Dokumente wurden schließlich mithilfe des Content-Management-Systems Drupal unter der Verwendung des XSL-Stylesheets in HTML konvertiert und in einem Webbrowser angezeigt.

Die erfolgreiche Realisierung des Projekts bildete die Grundlage für eine Weiterentwicklung des XSL-Stylesheets, deren Zielsetzung vor allem darin bestand, die Be-

[1] Das Ergebnis dieses Projekts ist auf der folgenden Seite zu finden: http://www.berardier.org/essai

dienbarkeit zu erleichtern. Seine Handhabung war insbesondere für Benutzer ohne Kenntnisse in Skriptsprachen problematisch, da gegebenenfalls eine Anpassung des Stylesheets per Hand hätte vorgenommen werden müssen. Als Beispiel sei in diesem Zusammenhang das Ändern des Farbschemas genannt, welches ohne eine manuelle Abänderung des Quellcodes nicht möglich war. Darüber hinaus war es erforderlich, die Template-Datei des in Drupal verwendeten Themes manuell zu erweitern, um das Javascript einbinden zu können.

Nach Abschluss des Editionsprojekts widmete sich der Kommilitone Roman Kominek in seiner Diplomarbeit der Aufgabe, ein Drupal-Modul zu erstellen, das eine automatisierte Einbindung des XSL-Stylesheets sowie des Javascripts in Drupal ermöglicht. So sollte im Speziellen die Visualisierung der TEI-Dokumente und die Anpassung der Darstellung vereinfacht werden, damit auf eine manuelle Änderung des Quellcodes verzichtet werden kann. Das entwickelte Modul erhielt schließlich den Namen „TEIChi“.

Die Zielsetzung der vorliegenden Arbeit besteht nun darin, die TEI-Dokumente in andere Formate wie EPUB oder Plain Text zu konvertieren und zum Herunterladen bereitzustellen, das genannte TEIChi-Modul ist hierfür konstitutiv.

1.2 Aufgabenstellung

Digitale Bücher, E-Books genannt (von engl. „electronic book“), werden immer beliebter, nicht nur im privaten Gebrauch, auch in Wissenschaft und Forschung. So haben sich E-Books als digitale Buchkopien vor allem bei Fachpublikationen eine Marktstellung sichern können. Ihr Vorteil liegt in Funktionen wie der Volltextsuche, die insbesondere bei Fachbüchern unverzichtbar geworden ist. Die Auswahl an Fachbüchern, die digital publiziert werden, wächst stetig. Auf der Frankfurter Buchmesse im Jahr 2007 waren bereits 30 Prozent aller Fachbücher als E-Books erhältlich [2]. Zudem sind zahlreiche historische Schriften, für die keine Urheberrechte mehr bestehen, als E-Books erschienen und damit frei zugänglich und sofort verfügbar.

Um der wachsenden Popularität der E-Books Rechnung zu tragen, besteht die Aufgabe der vorliegenden Diplomarbeit in der Entwicklung eines Drupal-Moduls, welches die TEI-Dokumente als EPUB-Version für E-Reader zum Download anbietet. Eine Plain Text- als auch die TEI/XML-Version sollen zusätzlich zum Herunterladen bereit-

gestellt werden. Ziel ist es, das Modul so zu konzipieren, dass die Wahl verschiedener Textvarianten (Originalfassung oder korrigierte Version) sowie eine Bestimmung des Textumfangs (Haupttext inklusive Anmerkungen des Autors *oder* Haupttext inklusive Anmerkungen des Autors und des Herausgebers) möglich sind. Eine Auswahlliste soll es dem Benutzer darüber hinaus erlauben, ausschließlich die Kapitel seiner Wahl herunterzuladen.

1.3 Aufbau der Arbeit

In der vorliegenden Arbeit werden zunächst einige wichtige Grundlagen erläutert, da sie das theoretische Fundament für die Entwicklung des Moduls bilden. Hierzu gehören der Aufbau und die Struktur von TEI-Dokumenten, das Content-Management-System Drupal sowie das EPUB-Format.

Im dritten Kapitel wird der konkrete Entwurf bzw. das Design des in dieser Arbeit zu entwickelnden Moduls skizziert.

Anschließend soll im vierten Kapitel die Verwendung, das heißt die spezifischen Funktionen des Moduls erörtert und mithilfe von Screenshots veranschaulicht werden.

Die Implementierung wird im fünften Kapitel anhand von Quelltext-Ausschnitten dargelegt, um einen etwas tieferen Einblick in die Programmiertechnik des konzipierten Drupal-Moduls vermitteln zu können.

Den Schlusspunkt dieser Arbeit bilden eine kurze Zusammenfassung der erzielten Ergebnisse sowie ein Ausblick auf mögliche Weiterentwicklungen des Moduls.

2 Grundlagen

An dieser Stelle werden einige Elemente erläutert, die die theoretische Grundlage für die Entwicklung des in dieser Arbeit thematisierten Drupal-Moduls bilden: Das Dateiformat TEI, das Content-Management-System Drupal und das EPUB-Format sollen im Folgenden definiert und veranschaulicht werden.

2.1 Text Encoding Initiative (TEI)

Die Text Encoding Initiative (TEI) wurde 1987 gegründet und ist eine internationale Interessengemeinschaft von Philologen, die es sich zum Ziel gesetzt hat, ein standardisiertes Dokumentenformat zur Kodierung und zum Austausch von Texten bereitzustellen und weiterzuentwickeln [3]. Die verschiedenen Arten von literarischem und linguistischem Text wie Protokolle, Gedichte, Dialoge oder Fragmente, können auf diese Weise elektronisch in einem einheitlichen Format erfasst werden, was die Arbeit insbesondere für Literaturwissenschaftler, Bibliotheken und Verlage erheblich erleichtert. Die TEI hat allgemeine Richtlinien für den gesamten Prozess des elektronischen Publizierens von Texten festgelegt, von der Dokumentenanalyse über die Digitalisierung und die Textauszeichnung bis hin zur Publikation [4]. Ein Vorteil von TEI besteht darin, dass es die Kodierung von Texten unabhängig von Betriebssystem oder proprietären Programmen der Textspeicherung (wie MS-Word) erlaubt.

Das gleichnamige Dokumentenformat TEI wurde seit 1988 auf der Basis von SGML entwickelt. Ein erster Entwurf P1 (P für engl. „proposal“: dt. Vorschlag) erschien 1990, die erweiterte und korrigierte Zwischenversion P2 (1992) wurde zwei Jahre später durch die ihrerseits überarbeitete Version P3 (1994) ersetzt. Diese stellt die erste nicht als Entwurf bezeichnete TEI-Version dar. Um der Entwicklung und Verbreitung von XML Rechnung tragen zu können, wurde im Jahr 2000 das TEI-Konsortium gegründet. Im Juni 2002 veröffentlichte dieses seine XML-kompatible Version P4 als auch die Version TEI-Lite, die einen reduzierten Umfang an Elementen aufweist, wodurch der Benutzer nicht mit der ganzen Komplexität von TEI konfrontiert wird; die in der vorliegenden Arbeit behandelten Dokumente sind in TEI-Lite realisiert. Die aktuelle Version P5 wurde seit 2005 entwickelt und am 1. November 2007 veröffentlicht [5].

2.1.1 Aufbau eines TEI-Dokuments

Ein TEI-Dokument besteht aus einem Kopf (<teiHeader>) und seinem Inhalt (<text>), die Grundstruktur sieht wie folgt aus [6, S. 163]:

```
<TEI>
  <teiHeader>
     <!-- Metadaten für das TEI-Dokument -->
   </teiHeader>
   <text>
     <!-- Inhalt des Textes -->
   </text>
</TEI>
```

Der TEI-Header ist das digitale Titelblatt einer elektronischen Edition, das als Minimum den Titel und Angaben über den Autor und den Herausgeber eines Textes beinhaltet. Zusätzlich können im Header aber auch Angaben über die Kodierungsprinzipien und weitere Zusatzinformationen gemacht werden, wie es im Folgenden zu sehen ist [6, S. 164]:

```
<teiHeader>
   <fileDesc>
     <!-- Bibliographische Beschreibung -->
   </fileDesc>
   <encodingDesc>
     <!-- Beschreibung der Kodierung-->
   </encodingDesc>
   <revisionDesc>
     <!-- Überarbeitungsschritte -->
   </revisionDesc>
</teiHeader>
```

Das Element <fileDesc> enthält die bibliographische Beschreibung eines elektronischen Textes, also den Titel, den Autor, die momentan vorliegende Ausgabe und die

Angabe des Verlags, bei dem der Text ursprünglich erschienen ist. Unter <encodingDesc> finden sich die Beschreibung der Kodierung zwischen dem elektronischen Text und seiner Quelle sowie eine detaillierte Beschreibung der bei der Kodierung des Textes angewendeten Editionsprinzipien und Verfahren. Das Element <revisionDesc> umfasst schließlich alle Überarbeitungsschritte, die an der Datei vorgenommen wurden.

Mit dem Element <text> wird hingegen in TEI der Dokumenteninhalt (Element <body>), die Informationen zur Titelseite oder Vorwort (Element <front>) als auch zum Anhang (Element <back>) ausgezeichnet [6, S. 167].

```
<text>
   <front>
      <!-- Vorwort, Inhaltsverzeichnis -->
   </front>
   <body>
      <!-- Dokumenteninhalt-->
   </body>
   <back>
      <!-- Anhang, Glossar, Index -->
   </back>
</text>
```

Für die Auszeichnung des Dokumenteninhalts stellt TEI Gliederungselemente in der Form divn zur Verfügung, wobei n die Gliederungsebene wiedergibt. Sollte die Gliederungshierarchie irrelevant sein, kann auf die Gliederungsebene verzichtet werden. Mithilfe des Attributs type ist es möglich, die Art der Gliederungsebene näher zu definieren: Für Gedichte werden Werte wie Strophe oder Vers, für Bücher Chapter oder Part verwendet. Darüber hinaus können Gliederungselemente mit dem Attribut xml:id versehen werden, wodurch diese eine eindeutige Bezeichnung erhalten. Jedes „divn“-Element beinhaltet zumeist ein Element head für den Titel oder die Überschrift sowie weitere Elemente für den Inhalt. Dieser wird wiederum in einzelne Absätze mit dem Element p gegliedert.

```
<body>
   <div type="chapter" xml:id="ENT01">
```

```
        <head>Titel oder Überschrift</head>
      <p> Erster Absatz  </p>
      <p> Zweiter Absatz  </p>
      ...
   </div>
</body>
```

Für die Darstellung eines Textes innerhalb eines Absatzes können u.a. die nachfolgenden Elemente benutzt werden:

- `<choice>` zum Erfassen zweier Textvarianten, z.B. die Originalfassung und die korrigierte Version

  ```
  <choice>
     <sic>Informatek</sic>
     <corr>Informatik</corr>
  </choice>
  ```

- `<lg>` zum Erfassen einer Strophe im Gedicht

  ```
  <lg>
     <l>Erste Zeile</l>
     <l>Zweite Zeile</l>
  </lg>
  ```

- `<pb n="xxx">` zur Angabe der Seitenzahl
- `<note>` zur Wiedergabe von Anmerkungen des Autors bzw. des Herausgebers

  ```
  <note resp="author">Anmerkung des Autors</note>
  <note resp="editor">Anmerkung des Herausgebers</note>
  ```

Das Attribut `rend`, das in jedem TEI-Element erlaubt ist, dient der Kennzeichnung von Textbestandteilen. Diese Information kann bei der Visualisierung genutzt werden, um Absätze oder einzelne Wörter mittels Fettschreibung bzw. Kursivschreibung hervorzuheben.

Die dieser Arbeit zugrunde liegenden TEI-Dokumente verwenden eine Teilmenge von Elementen des TEI-Lite-Standards.

2.2 Drupal

Drupal ist ein klassisches Content-Management-System (CMS), mit dem Inhalte auf einem Webserver veröffentlicht und bearbeitet werden können und dies auch ohne Programmier- und HTML-Kenntnisse. Es wurde in der Programmiersprache PHP geschrieben und verwendet MySQL oder PostgreSQL als Datenbank [7]. Vom belgischen Informatiker Dries Buytaert entwickelt, diente es an der Universität Antwerpen ursprünglich als Kommunikationsoberfläche für eine Gruppe von Studenten, die es ihnen ermöglichte, im Uni-Netzwerk über eine Website Nachrichten auszutauschen. Um nach Beendigung ihres Studiums weiterhin in Kontakt bleiben zu können, entschieden sich die Studenten, das von Buytaert konzipierte System online unter dem Domainnamen „dorp.org“ („Dorp“ ist niederländisch für „Dorf“) fortzuführen. Ein Tippfehler bei der Anmeldung der Domain und die ohnehin fehlende Verfügbarkeit des gewünschten Domainnamens, hatten zur Folge, dass die Software schließlich unter „drop.org“ im Netz veröffentlicht wurde. „Drop“ (deutsch: „Tropfen“) heißt auf Niederländisch „Druppel“ („drüppel“ gesprochen). Der heute bekannte Name Drupal stellt die Lautschriftvariante dieses Wortes dar, um auch im englischsprachigen Raum eine korrekte niederländische Aussprache sicherzustellen [8, S. 30f.].

Seit 2001 ist Drupal Open Source, das heißt eine frei verfügbare und kostenlose Software, die der GNU General Public License untersteht. Drupal ist zudem eine „Social Software“ mit einer großen Community von Benutzern, Administratoren, Webdesignern und -entwicklern, die Hilfestellung beim Erlernen und bei der Installation des Systems gibt und darüber hinaus neue Module und Themes konzipiert, so dass seine Aktualität als auch eine Fehlerbehebung ständig gewährleistet sind. Drupal 6.19 ist zurzeit die aktuelle stabile und getestete Version (Stand vom 19.12.2010), die vorliegende Arbeit bezieht sich jedoch auf die sich noch in der Entwicklung befindende Version 7.0.

Der große Vorteil von Drupal liegt in seiner Flexibilität; die modulare Erweiterbarkeit des Systems nach individuellen Anforderungen ermöglicht eine schnelle und effiziente Realisierung von Webinhalten, wodurch es sich neben anderen Content-Management-Systemen wie Joomla! [9] und TYPO3 [10] erfolgreich etablieren konnte. Auch Aspekte wie Sicherheit, Wartbarkeit und Geschwindigkeit finden ihre Berücksichtigung: Ein aus Experten bestehendes „Security-Team“ nimmt sich der Sicherheit von Drupal an. Es verwundert daher nicht, dass Drupal weltweit für verschiedenste

Websites zum Einsatz kommt, unter anderem auch für die Webpräsenz des Weißen Hauses in Washington [7].

2.2.1 Aufbau und Funktionen

Drupal ist so konzipiert, dass es auf allen Betriebssystemen verwendet werden kann, die eine PHP-Unterstützung aufweisen. Als Webserver kommt zumeist Apache zum Einsatz, IIS von Microsoft u.a. sind aber ebenfalls möglich. Wie oben bereits erwähnt, werden in der Regel MySQL und PostgreSQL als Datenbanken eingesetzt. Da Drupal eine Datenbankabstraktionsschicht besitzt, können jedoch auch andere SQL-Datenbanken verwendet werden, so dass keine neuen Datenbankabfragen vorgenommen werden müssen.

Drupal selbst basiert auf einem Systemkern (engl. „Core“), der die Grundfunktionalität gewährleistet. Er umfasst die sogenannten Kern-Module, die sich in zwei Arten gliedern lassen: Zum einen gibt es Systemkern-Pflichtmodule, die für den Betrieb von Drupal erforderlich sind und nicht deaktiviert werden können. Die optionalen Systemkern-Module hingegen beinhalten Funktionen wie Blogs und Foren, die nach Bedarf aktiviert bzw. deaktiviert werden können [8, S. 165]. Zu den Systemkern-Pflichtmodulen zählen:

- **System-Modul**: Dieses Modul ist die Basis jeder Drupal-Website. Es gewährleistet die Grundfunktionalität wie das Caching, die Aktivierung bzw. Deaktivierung von Modulen und Themes, die Verwaltung als auch die Konfiguration der Website.
- **User-Modul**: Es ermöglicht Benutzern die Registrierung, das Anmelden bzw. Abmelden sowie die Rollenverwaltung. In Drupal gibt es zwei Rollen, die des authentifizierten (angemeldeten) und die des anonymen (nicht angemeldeten) Benutzers.
- **Node-Modul**: Sämtliche von Drupal verwaltete Inhalte werden in sogenannten *Nodes* abgespeichert. Der Node ist ein strukturierter Inhaltstyp, z.B. eine Buch-, Blog- oder Produktseite; er besitzt eine eindeutige ID (Identifikationsnummer), die sich bei jedem neu erstellten Inhalt um eins erhöht. Jeder Node verfügt grundsätzlich über einen Titel und einen Inhalt. Standardmäßig gibt es bei Drupal zwei Inhaltstypen, „Story“ (Artikel) und „Page“ (Seite): Der Inhaltstyp „Sto-

ry“ findet seine Anwendung vorwiegend bei der Darstellung dynamischer Inhalte, die oft verändert, kommentiert oder aktualisiert werden wie z.B. Forumsbeiträge und Pressemitteilungen. Mit dem Inhaltstyp „Page“ dagegen werden vorwiegend statische Inhalte realisiert; er wird für Websites verwendet, die sich nur selten ändern und die auf die Möglichkeit der Interaktion zwischen Inhalt und Benutzer verzichten; Beispiele hierfür sind das Impressum oder die AGBs (Allgemeinen Geschäftsbedingungen) einer Website [11].

- **Block-Modul**: Mithilfe dieses Moduls können Blöcke erstellt werden, die der Platzierung von Daten auf einer Website dienen. Als „Blöcke“ werden Daten und Funktionalitäten bezeichnet, z.B. Navigationsblöcke, Log-In-Blöcke und Download-Blöcke, die zumeist im Randbereich einer Website zu finden sind, z.B. in der linken oder rechten Sidebar. Blöcke besitzen einen Titel und eine Beschreibung, verfügen aber über keinen konkreten Inhalt, was sie somit von den Nodes unterscheidet. Die Bereiche, in denen Blöcke platziert werden, nennen sich „Regionen“. Das Zuordnen eines Blocks zu einer Region wird in der Verwaltungsoberfläche von Drupal vorgenommen. Geschieht dies nicht, wird dieser auch nicht angezeigt. Es ist möglich, Blöcke so einzurichten, dass sie nur auf bestimmten Seiten oder in einem speziellen Kontext dargestellt werden [12]. Der in dieser Arbeit thematisierte Download-Bereich wird als Block realisiert.
- **Filter-Modul**: Dieses Modul filtert die Benutzereingaben, bevor sie in der Datenbank abgespeichert und auf der Website angezeigt werden, das heißt, es werden unerwünschte Formatierungen wie HTML, PHP, JavaScript o.Ä. aus eingegebenen Texten entfernt.

Und zu den optionalen Systemkern-Modulen [11] gehören u.a.:

- **Book-Modul**: Mit diesem Modul können Inhalte (Nodes) hierarchisch gegliedert (kapitelweise, nach Abschnitten o.ä.) und als Buch zusammengefasst werden. Dies bietet sich insbesondere für Seiteninhalte an, die über eine Buchstruktur verfügen. Das Modul hält darüber hinaus die Möglichkeit bereit, einen Navigationsblock, eine Druckansicht des Inhalts sowie Links zum Vor- und Zurückblättern zu erstellen.

- **Menu-Modul**: Mithilfe dieses Moduls kann das Menüsystem von Drupal gesteuert und angepasst werden. Menüs werden in diesem Kontext als Sammlungen von Links definiert, welche der Navigation auf der Website dienen.
- **Search-Modul**: Dieses Modul ermöglicht eine Volltextsuche über den gesamten Inhalt der Website.
- **Comment-Modul**: Es dient dem Kommentieren von Inhalten einer Website, insbesondere bei Forenthemen, Blogbeiträgen usw.
- **Help-Modul**: Es unterstützt und hilft bei der Verwendung und Konfiguration von Drupal und seiner Module.

Über die Systemkern-Module hinaus gibt es derzeit mehr als 7.330 Module, sogenannte „add-ons“ (Erweiterungen), die von der Drupal-Community entwickelt und auf drupal.org veröffentlicht wurden. Davon sind etwa 5.080 kompatibel zu Drupal 6, für das (zum Zeitpunkt der vorliegenden Arbeit) noch ausstehende Drupal 7 sind bereits 771 Module konzipiert worden [13].

Zu guter Letzt seien „Themes“ erwähnt, Sammlungen von Dateien, die in Drupal für das Layout einer Website zuständig sind [14, S. 215]. Sie sind zum Teil schon in der Grundeinstellung enthalten und damit dem Systemkern von Drupal zugehörig, aber es gibt auch Themes, die von der Drupal-Community erarbeitet wurden. Zurzeit stehen etwa 895 Themes auf der Drupal-Homepage zum Download bereit.

Die nachfolgende Abbildung stellt den Systemaufbau von Drupal vereinfacht dar.

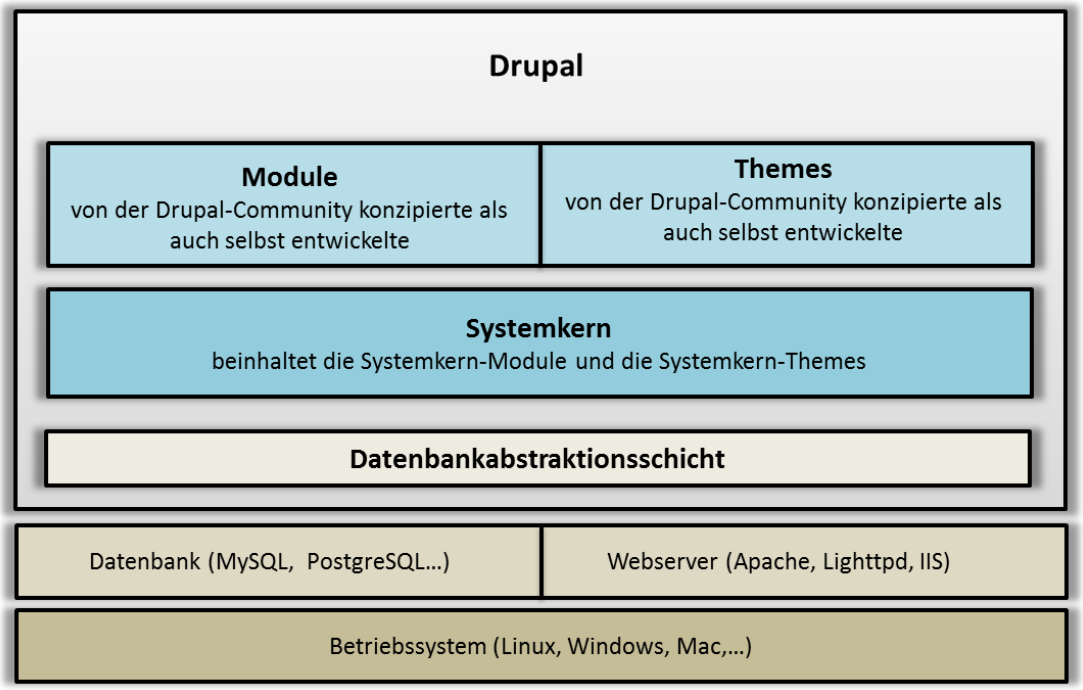

Abb. 2-1 Systemarchitektur von Drupal, erweitert nach VanDyk **[14, S. 34]**

2.2.2 Hook-System

Drupal basiert auf einem System von Hooks, „Hook“ ist das englische Wort für „Haken“ bzw. „Greifer“. Hooks sind Software-Schnittstellen (API), die es erlauben, ein selbst entwickeltes Modul mit dem Grundsystem von Drupal interagieren zu lassen. Funktionen eigener Module können sich somit in spezifische Stellen des Systems „einhaken“ und dort die gewünschten Operationen ausführen. Ein Hook ist eine PHP-Funktion mit dem Namen `modul_hook()`, wobei „modul“ der Name des Moduls und „hook“ der Name des Hooks ist. Jeder Hook besitzt eine Reihe von Eingabeparametern und ein Ergebnis als Rückgabewert [15].

2.3 EPUB-Format

EPUB steht für „electronic publication“ und ist ein offenes, standardisiertes Dateiformat für die Darstellung von Dokumenten auf E-Book-Readern. Aber auch auf Smartphones und Programmen unter Windows, Mac-OS und Linux lässt sich das EPUB-Format anzeigen. Entwickelt wurde es vom International Digital Publishing Forum (IDPF) und

von diesem im Jahr 2007 veröffentlicht. Das IDPF [16] ist eine US-amerikanische Branchenorganisation von Verlagen, Format- und Geräteherstellern, deren Ziel die Standardisierung, Weiterentwicklung und Förderung des elektronischen Publizierens ist. Im Vergleich zu PDF-Dateien erlaubt EPUB eine dynamische Anpassung des Textinhalts an die verschiedenen Bildschirmgrößen, was insbesondere bei kleinen Displays von Vorteil ist: Bei einer Vergrößerung des Textes auf dem Lesegerät brechen die Textzeilen automatisch um, ein horizontales Scrolling ist somit nicht mehr erforderlich. Zudem bietet das EPUB-Format die Möglichkeit einer buchähnlichen Verarbeitung von inhaltlich formalen Bestandteilen eines Textes wie Kapitel, Grafiken und Fußnoten als auch ein verlinktes Inhaltsverzeichnis. Ferner sei die Unterstützung von Digital Rights Management-Mechanismen (DRM) erwähnt, welche die Verwendung und Verbreitung digitaler Medien kontrollieren und hiermit eine digitale Rechteverwaltung gewährleisten. Konkret bedeutet dies, dass EPUB-Bücher individuell verschlüsselt und gegen nicht berechtigte Vervielfältigung geschützt werden können [17, S. 10f.].

2.3.1 Aufbau einer EPUB-Datei

Das EPUB-Format basiert hauptsächlich auf XML und XHTML und setzt sich aus den drei folgenden offenen Standards zusammen [18]:

- Open Publication Structure (OPS), für die eigentlichen Inhaltsdaten von EPUB-Dokumenten
- Open Packaging Format (OPF), für die Beschreibung der Struktur der EPUB-Datei und die Aufnahme der Metadaten
- Open Container Format (OCF), fasst die Dateien als ZIP-Archiv mit der Dateiendung .epub zusammen

Eine EPUB-Datei ist also ein speziell präpariertes ZIP-Archiv, das eine feste Verzeichnisstruktur sowie eine Reihe von Steuerdateien und die eigentlichen Inhaltsdokumente enthält. Das Grundgerüst einer EPUB-Datei sieht hierbei immer gleich aus. Mithilfe der nachfolgenden Abbildung sollen die wichtigsten Elemente einer EPUB-Datei erläutert werden.

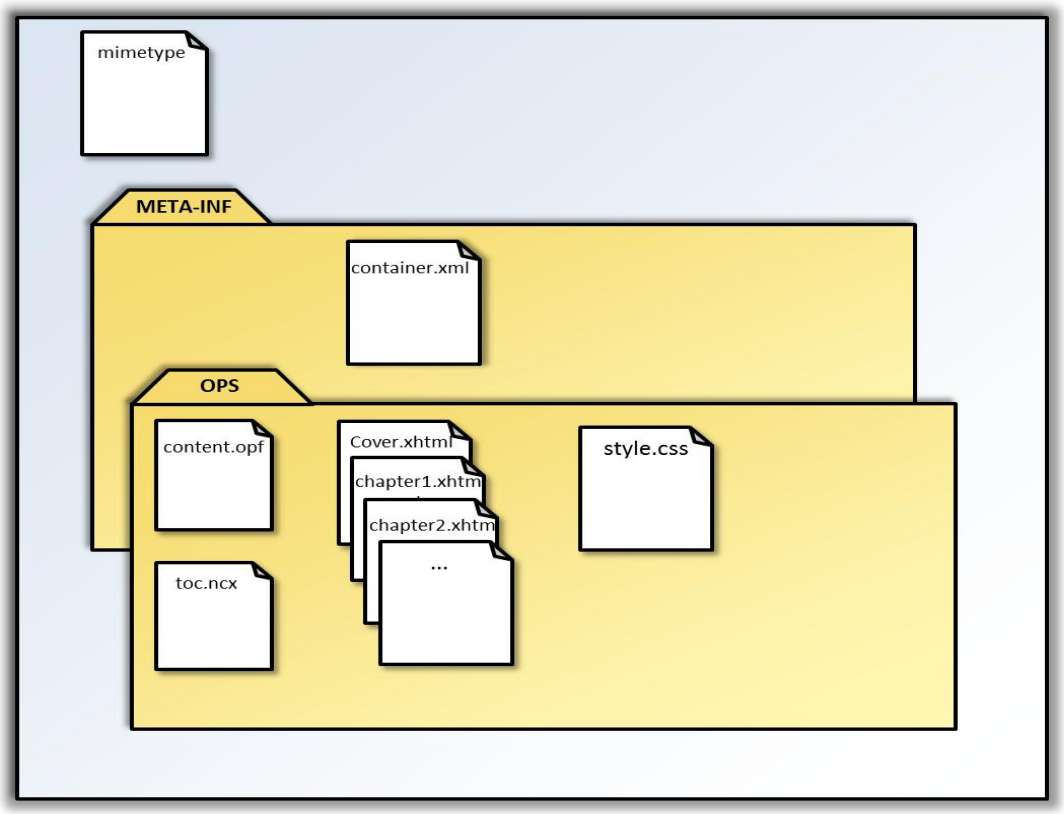

Abb. 2-2 Der Aufbau einer EPUB-Datei erweitert nach Helder **[19]**

Die Datei „mimetype“ ist bei allen EPUB-Dateien immer gleich und enthält ausschließlich die Zeile „`application/epub+zip`“, die den Typ des gesamten Archives festlegt. Sie befindet sich immer an der ersten Stelle des Archives und teilt den Lesegeräten oder der Lesesoftware mit, dass es sich hier um eine EPUB-Datei im ZIP-Format handelt [20].

Es folgt das Verzeichnis META-INF, welches – wie die Benennung vermuten lässt – die Meta-Informationen zu den Inhalten der EPUB-Datei beinhaltet. In diesem Verzeichnis liegt die Datei „container.xml“ sowie andere optionale Dateien für Signaturen, Verschlüsselung und digitales Rechtemanagement. Die Datei container.xml ist jedoch die wichtigste und *muss* vorhanden sein. Sie ist im XML-Format und folgendermaßen aufgebaut:

```
<?xml version="1.0" encoding="UTF-8"?>
<container version="1.0"
  xmlns="urn:oasis:names:tc:opendocument:xmlns:container">
   <rootfiles>
      <rootfile full-path="OPS/content.opf"
```

```
         media-type="application/oebps-package+xml"/>
   </rootfiles>
</container>
```

Das Attribut `full-path` verweist auf die jeweilige OPF-Datei der Publikation. Diese ist im OPS-Verzeichnis zu finden. Der folgende Code soll den Aufbau der Datei veranschaulichen [21]:

```
<?xml version="1.0" encoding="UTF-8"?>
<package version="2.0"
  xmlns="http://www.idpf.org/2007/opf"
     unique-identifier="EPB-UUID">
   <metadata>
      <dc:title></dc:title>
      <dc:creator opf:role="aut"></dc:creator>
      <dc:publisher></dc:publisher>
      ...
   </metadata>
   <manifest>
      <!-- Content Documents -->
      <item id="ID_01" href="chapter_1.xhtml"
          media-type="mime-type">
      <item id="ID_02" href="chapter_2.xhtml"
          media-type="mime-type">
      <!-- CSS Style Sheets -->
      <item href="style.css" media-type="text/css"/>
      <!-- NCX -->
      <item href="toc.ncx"
           media-type="application/x-dtbncx+xml"/>
   </manifest>
   <spine toc="ncx">
      <itemref idref="ID_01"/>
      <itemref idref="ID_02"/>
   </spine>
</package>
```

Das Element <metadata> enthält die wichtigsten Eckdaten eines E-Books, das heißt den Titel, den Autor und Herausgeber, den Ort der Publikation usw. Innerhalb des Elements <manifest> werden alle für das konkrete E-Book benötigten Dateien aufgelistet. Hierzu gehören die Kapitel des Buches, die Formatierungsanweisungen (CSS-Datei) als auch die NCX-Datei. Mit dem Element <spine> hingegen wird die Reihenfolge für die Darstellung der einzelnen Inhaltsdokumente statuiert (z.B. Kapitel 1, Kapitel 2, Kapitel 3 usw.).

Die toc.ncx-Datei gibt das Inhaltsverzeichnis des Buches wieder. Die einzelnen Kapitel werden hier in einer Navigation <navMap> abgespeichert.

```
<navMap>
      <navPoint playOrder="1">
          <navLabel>
              <text>Kapitel 1</text>
          </navLabel>
          <content src="chapter_1.html"/>
       </navPoint>
      <navPoint playOrder="2">
          <navLabel>
              <text>Kapitel 2</text>
          </navLabel>
          <content src="chapter_2.html"/>
       </navPoint>
</navMap>
```

Das Attribut playOrder gibt die Reihenfolge an, in der die einzelnen Navigationspunkte aufgeführt werden sollen. Mit dem Element <navLabel> wird der Text, der in der Navigationsleiste angezeigt ist, festgelegt. Eine Referenz zum jeweiligen Kapitel wird schließlich mit dem Element <content> erstellt [22].

In der Datei style.css können eigene Textstrukturierungselemente bestimmt werden wie die Schriftart und -größe als auch die Positionierung, z.B. die Ausrichtung des Textes (linksbündig, zentriert, rechtsbündig oder Blocksatz), die Anordnung der Überschriften und der Abstand zum Seitenrand.

3 Entwurf des Moduls

Im Folgenden wird der Entwurf des zu implementierenden Moduls skizziert. Aus der Zielsetzung – die Entwicklung eines Drupal-Moduls, welches die TEI-Dokumente als EPUB-Version zum Download bereitstellt – resultiert der für die Implementierung obligatorische Name: TEI2Pub-Modul. Die Endung „2Pub“ leitet sich hierbei aus dem Englischen „to publish“ (dt. „veröffentlichen“) her. Die Festlegung des Modulnamens in der Entwurfsphase ist sehr wichtig, da alle später zu implementierenden Funktionen mit dem Namen des Moduls beginnen. Hierbei ist zu berücksichtigen, dass der Modulname einzigartig zu sein hat, kein anderes Modul bzw. Theme darf den gleichen Namen tragen [23, S. 319].

Nachdem die „Namensgebung“ des Moduls nun als abgeschlossen angesehen werden kann, sollen nachfolgend die wichtigsten Elemente des Entwurfs dargestellt werden.

3.1 Block-Modul

Um Flexibilität gewährleisten zu können, soll der erwähnte Download-Bereich als Block-Modul realisiert werden. Auf diese Weise ist seine Positionierung variabel und kann vom Administrator jederzeit neu angepasst werden. Der Block soll zunächst drei Icons für den direkten Download des aktuell angezeigten Buchkapitels beinhalten sowie über die Möglichkeit eines erweiterten Downloads verfügen. Letzterer soll dem Benutzer Auswahlmöglichkeiten hinsichtlich der Textvarianten und des Textumfangs bieten. Darüber hinaus wird ein Block-Modul angestrebt, das nicht auf allen in Drupal erstellten Seiten erscheint, sondern ausschließlich auf Seiten, deren Inhalt ein visualisiertes TEI-Dokument ist.

3.2 Konfigurationsmöglichkeiten

Ziel ist die Konzeption einer Konfigurationsseite im administrativen Bereich von Drupal, die eine Gestaltung des Blocks sowie zahlreiche Einstellungsoptionen für die EPUB-Version der in Drupal erstellten Bücher zur Verfügung stellt. Hierzu gehört z.B.

die Auswahl der Metadaten, die auf den ersten beiden Seiten des E-Books angezeigt werden. Der Administrator soll zudem die Möglichkeit besitzen, ein Titelbild hochzuladen und es dem entsprechenden Buch zuzuweisen.

3.3 Abhängigkeit des Moduls

Das TEIChi-Modul, wie bereits zu Beginn der Arbeit erwähnt, ist konstitutiv für den Entwurf des TEI2Pub-Moduls. Es besitzt eine Funktion mit dem Namen `teichi_get_teichi_nodes`, die eine Liste von visualisierten TEI-Dokumenten zurückgibt. Anhand dieser Liste kann überprüft werden, ob es sich bei dem Inhalt einer Seite um ein TEI-Dokument handelt. Ist dies der Fall, wird der Download-Bereich des TEI2Pub-Moduls angezeigt.

Zudem besitzt das TEIChi-Modul Funktionen, die auch das TEI2Pub-Modul nutzen wird, wie z.B. die Funktion `_teichi_get_book_for_node`. Diese Funktion gibt die Buch-ID zu einem bestimmten Kapitel zurück.

Die Abhängigkeit beruht also auf dem Verwenden von Funktionen, die das TEIChi-Modul bereits implementiert hat.

3.4 Programmierstandards von Drupal

Da das in dieser Arbeit konzipierte TEI2Pub-Modul zu einem späteren Zeitpunkt in der Drupal-Community veröffentlicht werden soll, ist die Einhaltung ihrer Coding Standards [24] unerlässlich. Ein „standardisiertes Erscheinungsbild“ der Codebasis verbessert nicht nur die Lesbarkeit, sie erleichtert darüber hinaus auch den Einstieg für neue Entwickler.

Aus diesem Grund werden beispielsweise für einen Einzug zwei Leerzeichen verwendet und kein Tabulator.

Der Konvention nach sind alle Funktionsnamen zudem in Kleinbuchstaben geschrieben und beginnen mit dem Namen des Moduls, in diesem Fall „tei2pub“. Deskriptive Teile des Funktionsnamens werden durch Unterstriche voneinander getrennt, z.B. `tei2pub_download_chapter_as_epub`.

Bei Arrays wird jedes einzelne Element in eine eigene Zeile geschrieben, somit können Arrayelemente problemlos beigefügt oder aber gelöscht werden [14, S. 49], es folgt ein Beispiel:

```
Form[text]  =  array(
  '#type' => 'textfield',
  '#title' => 'Text Field',
  '#size' => 30,
);
```

4 Verwendung

In diesem Kapitel soll der Umgang mit dem TEI2Pub-Modul veranschaulicht werden. Zunächst wird die Installation und Aktivierung des Moduls näher beschrieben, anschließend sollen seine spezifischen Konfigurations- und Verwendungsmöglichkeiten erläutert werden. Die eigentliche Implementierung der Funktionalitäten dieses Moduls wird in Kapitel 5 dargestellt.

4.1 Installation und Aktivierung des Moduls

Eine erfolgreiche Installation bildet die Grundlage für die Aktivierung der einzelnen Funktionen dieses Moduls. Zunächst muss es in das Modulverzeichnis von Drupal kopiert werden. Drupal stellt zwei Verzeichnisse für die Installation des Moduls zur Verfügung, die sich im Drupal-Installationsordner unter `…/modules` und unter `…/sites/all/modules` befinden. Für gewöhnlich werden selbst entwickelte oder von der Drupal Community konzipierte Module in das letztere Verzeichnis kopiert, im erstgenannten liegen die Systemkern-Module (vgl. Kapitel 2 dieser Arbeit).

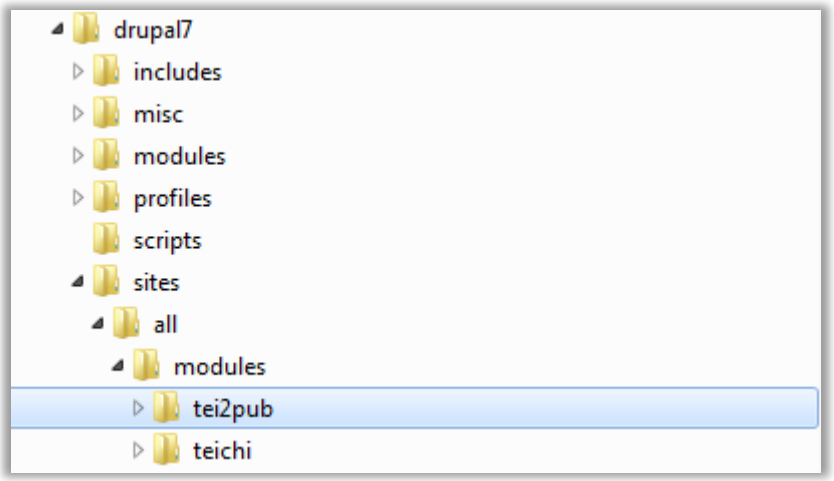

Abb. 4-1 Ordnerstruktur des Drupal-Installationsordners

Nachdem das Modul in Drupal eingebunden wurde, erscheint es nun im Administrationsmenü unter „modules“ (siehe Abb. 4-2). Hier sind eine Auflistung aller vorhandenen Module und ihr jeweiliger Status (aktiviert oder deaktiviert) zu finden.

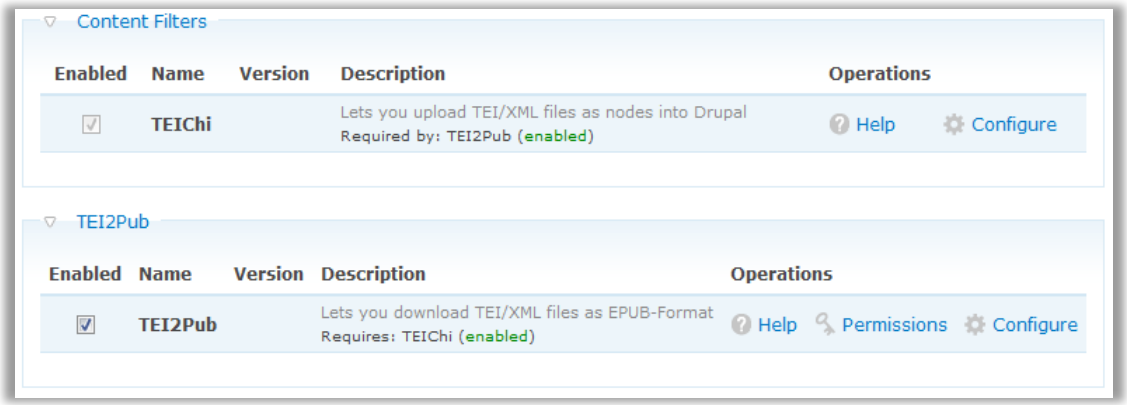

Abb. 4-2 Aktivierung des TEI2Pub-Moduls

Da das TEI2Pub-Modul eine Abhängigkeit zum TEIChi-Modul aufweist, muss dieses bereits aktiv sein oder aber zusammen mit ihm aktiviert werden. Wurde die Aktivierung fehlerfrei abgeschlossen, erscheinen drei Operationen: „Help“ bietet Hilfestellung und allgemeine Informationen zum TEI2Pub-Modul, die Operation „Permissions“ erlaubt die Verwaltung der Zugriffsrechte und mithilfe der Operation „Configure“ können diverse Konfigurationen des Moduls vorgenommen werden. Das TEI2Pub-Modul ist somit betriebsbereit und seine Konfiguration möglich.

4.2 Konfiguration

Rechtevergabe

Wie oben bereits erwähnt, bietet das TEI2Pub-Modul dem Benutzer zahlreiche Konfigurationsmöglichkeiten. Da es sich um eine grundlegende Einstellung handelt, soll hier zunächst die Konfiguration der Zugriffsrechte ihre Betrachtung finden. Die Anwahl der Operation „Permissions“ führt den Benutzer auf die Administrationsseite der Rechtevergabe, auf der rollenspezifische Berechtigungen erteilt werden. Auf diese Weise gewährleistet das Modul Sicherheit im Umgang mit vertraulichen Daten, nur berechtigten Usern ist das Herunterladen von Dokumenten erlaubt. Wie in Abb. 4-3 zu sehen ist, gibt es drei Benutzerrollen, für die verschiedene Zugriffsrechte gesetzt werden können: anonym, authentifiziert und Administrator.

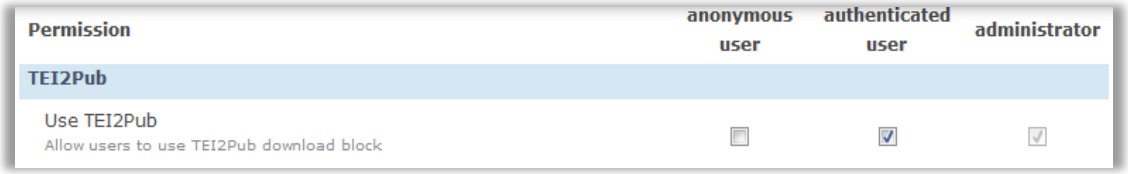

Abb. 4-3 Permission-Menü

Blockpositionierung

Da es sich bei dem hier thematisierten TEI2Pub-Modul um ein Block-Modul handelt, muss es auf der Blockadministrationsseite aktiviert und positioniert werden. Die Blockadministrationsseite ist im Administrationsmenü unter „structure → block“ zu finden. Mithilfe einer „Drag-and-Drop-Oberfläche“[2] können hier Blöcke einer Region zugewiesen und ihre Reihenfolge innerhalb dieser festgelegt werden. An dieser Stelle sei darauf hingewiesen, dass nicht alle Themes die gleichen Regionen verwenden oder diese in der gleichen Weise anzeigen. Aus diesem Grund werden Blöcke in Abhängigkeit zum aktuell eingestellten Theme angeordnet. Abb. 4-4 zeigt die verfügbaren Regionen, die für einen Block unter der Verwendung des Themes „Garland“ ausgewählt werden können.

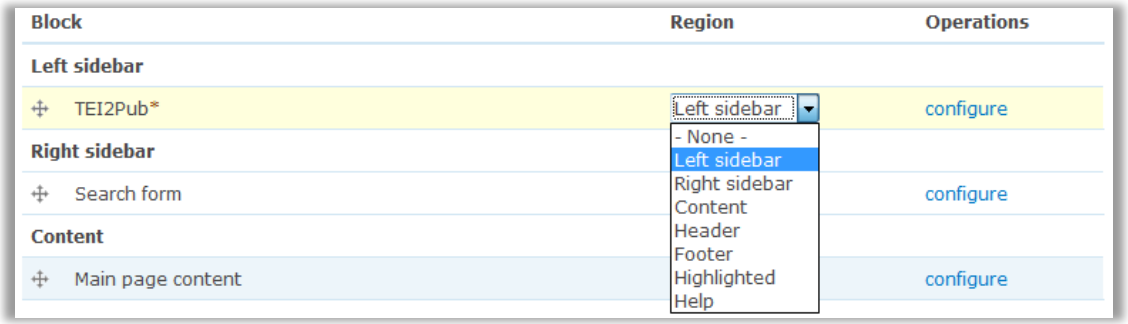

Abb. 4-4 Positionierungsmöglichkeiten des TEI2Pub-Blocks im Theme „Garland“

Nachdem die Region für den Block bestimmt wurde, lassen sich nun weitere Konfigurationen mittels der Operation „configure“ an diesem vornehmen. Der Block kann beispielsweise mit einem Titel versehen werden, um ihn benutzerfreundlicher zu gestalten (siehe Abb. 4-6, 4-7). Zudem lassen sich die Regionen für alle aktivierten Themes

[2] „Drag and Drop“ bedeutet auf Deutsch „Ziehen und Fallenlassen“ und stellt eine Methode zur Bedienung grafischer Benutzeroberflächen durch das Bewegen grafischer Elemente mittels eines Zeigegerätes (z.B. mit dem Mauszeiger) dar. So kann ein Element gezogen und über einem möglichen Ziel losgelassen werden (vgl. [33]).

gleichzeitig festlegen, so dass die Anzeige des Blocks auch bei einem Theme-Wechsel gewährleistet ist.

Abb. 4-5 Konfiguration des Blocks

Die folgenden Abbildungen zeigen den TEI2Pub-Block ohne und mit Titel.

Abb. 4-6 TEI2Pub-Block ohne Titel

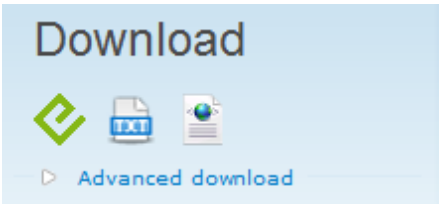

Abb. 4-7 TEI2Pub-Block mit Titel

Einstellung der Beschriftung im Download-Block

Das TEI2Pub-Modul bietet dem Benutzer eine Konfigurationsseite, auf der spezielle Einstellungen vorgenommen werden können. Diese kann im Administrationsmenü unter „Configuration“ aufgerufen werden als auch durch die Operation „Configure“ nach der

Aktivierung des Moduls (vgl. Abb. 4-2). Auf der Hauptseite der Modulkonfiguration werden dem Benutzer zwei Einstellungsmöglichkeiten bereitgestellt (siehe Abb. 4-8).

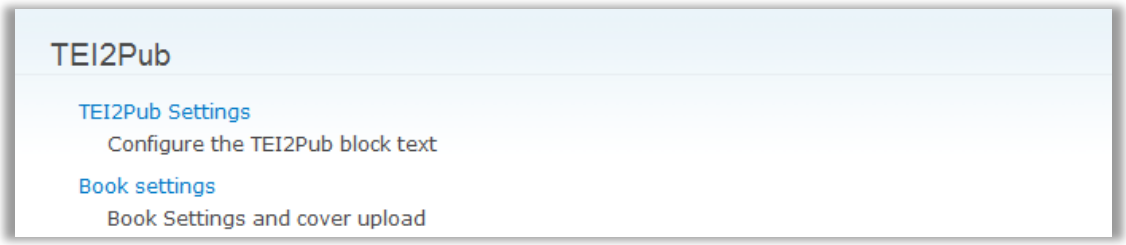

Abb. 4-8 Hauptseite der Konfiguration des TEI2Pub-Moduls

Mit „TEI2Pub Settings“ lassen sich die Beschriftungen aller Optionen des im Block angezeigten Download-Formulars modifizieren. Über die Einstellung „Book Settings“ hingegen kann ein Buch den individuellen Wünschen hinsichtlich der Art des Downloads und der Buchgestaltung angepasst werden. Zudem ist in diese Einstellung der Upload und die Zuweisung von Titelbildern zu dem entsprechenden Buch integriert. Wird „TEI2Pub Settings“ nun angewählt, öffnet sich die folgende Konfigurationsseite:

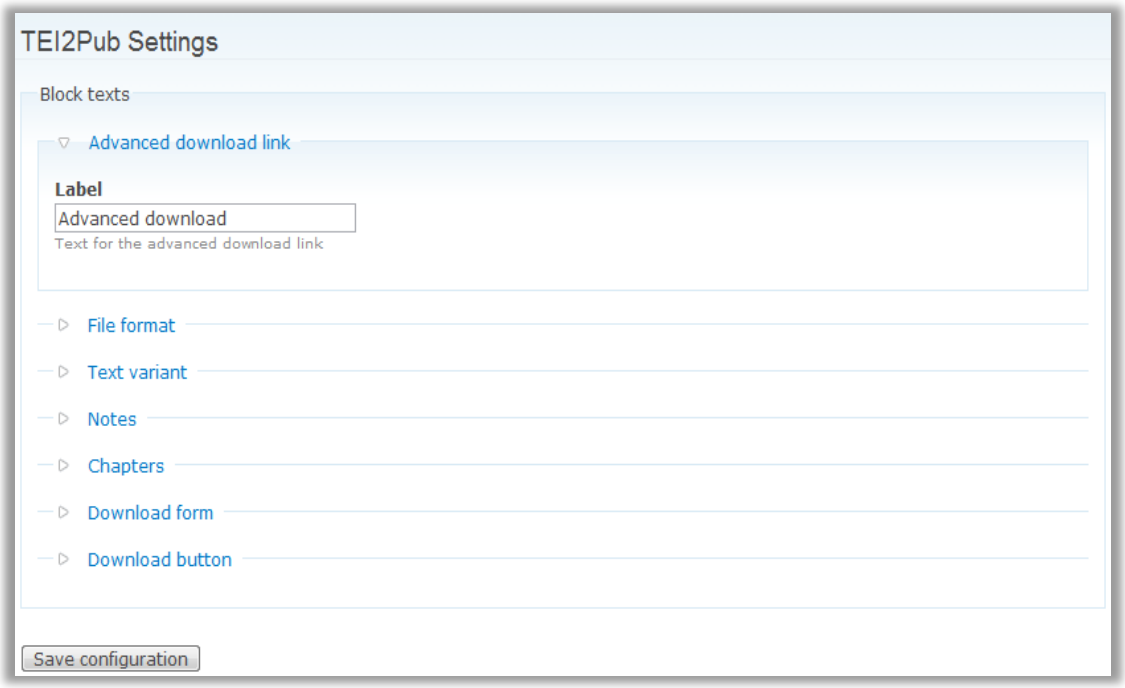

Abb. 4-9 Einstellung der Beschriftungen im Download-Formular

Die in Abb. 4-9 zu sehende Konfigurationsseite besteht aus mehreren aufklappbaren Menüs. Sie sind in der Reihenfolge angeordnet, in der sie im Download-Formular angezeigt werden. Beispielsweise kann der in Abb. 4-6 und Abb. 4-7 gezeigte „Advanced download"-Link in „Erweiterter Download" umbenannt oder je nach Wunsch auch in andere Sprachen übersetzt werden. Drupal stellt zwar eine Übersetzungsfunktion für die Benutzeroberfläche zur Verfügung, an dieser Stelle wurde jedoch auf ihre Verwendung verzichtet, um irreführende bzw. nicht sinngemäße Übersetzungen zu vermeiden. So hat der Benutzer die Möglichkeit, die Beschriftung selbst zu bestimmen.

Wurden alle notwendigen Einstellungen vorgenommen, können diese anschließend mithilfe des „Save configuration"-Knopfes gespeichert werden.

Bucheinstellungen

Wenn die zweite Einstellungsmöglichkeit der Modulkonfiguration „Book settings" angewählt wird, öffnet sich das folgende Fenster:

Abb. 4-10 Bücherkonfiguration

In der Tabelle werden alle erstellten Bücher aufgelistet. Jedes Buch lässt sich separat konfigurieren. Über das Menü „Configure settings" können zunächst die Dateiformate für den Download des E-Books bestimmt werden. In der Standard-Einstellung sind die nachfolgenden drei Formate aktiviert: EPUB, Plain Text und TEI/XML (siehe Abb. 4-11). Mithilfe von Auswahlkästchen (Checkboxes) kann der Administrator nun das gewünschte Dateiformat festlegen. So ist es ihm möglich, dem Benutzer beispielsweise nur das EPUB-Format zum Download bereitzustellen oder aber eine beliebige Kombination der genannten Dateiformate, es muss jedoch wenigstens eines der Formate ausgewählt sein. Die aktivierten Formate werden anschließend im Download-Block mit dem entsprechenden Symbol angezeigt (wie in Abb. 4-6 und Abb. 4-7 zu sehen). Es sei an dieser Stelle darauf hingewiesen, dass die verwendeten Symbole im TEI2Pub-

Modulverzeichnis im Unterverzeichnis „images“ zu finden sind. Hier können die Symbole je nach Bedarf auch durch andere ersetzt werden

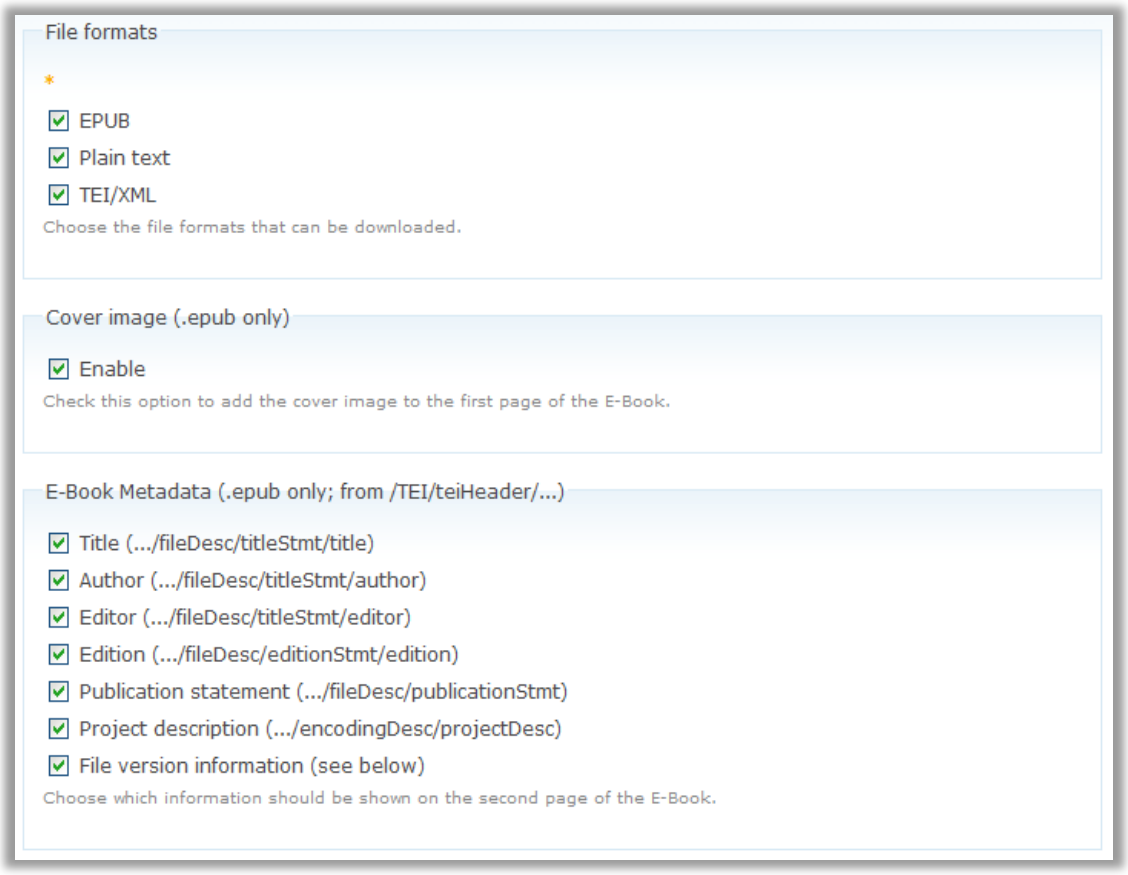

Abb. 4-11 Konfigurationsmöglichkeiten für den Download eines E-Books

Neben dem Dateiformat kann der Administrator ebenfalls mithilfe von Checkboxes die Einstellungen für die ersten beiden Seiten des E-Books bestimmen. Zum einen kann er festlegen, ob auf der ersten Seite ein Titelbild angezeigt wird oder nicht.

Für die zweite Seite des E-Books besteht zudem die Wahl zwischen verschiedenen Metadaten. Die Metadaten selbst werden automatisch dem TEI-Header (vgl. Kapitel 2.3) entnommen. Für die Informationen zur Dateiversion stehen Eingabefelder zur Verfügung, mit deren Hilfe der Administrator ihre Angaben bestimmen kann. Die Anzeige der Metadaten auf einem E-Book-Reader könnte folgendermaßen aussehen:

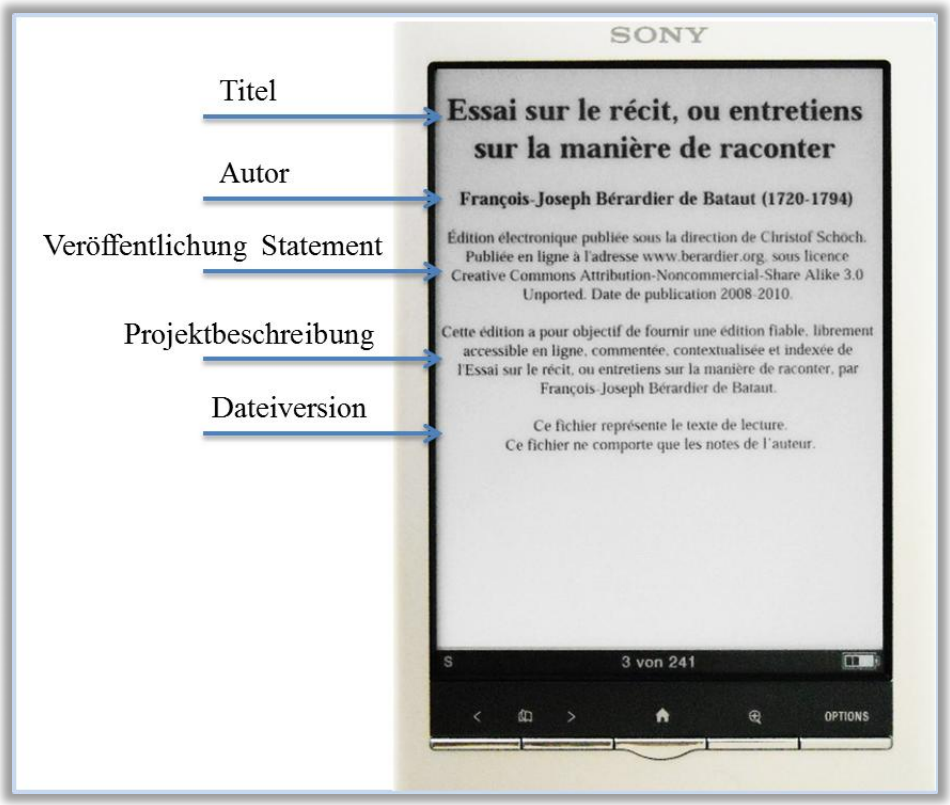

Abb. 4-12 Anzeige der Metadaten auf einem SONY E-Reader PRS-650

Soll kein Titelbild angezeigt werden, erscheinen die Metadaten auf der ersten Seite. Sind keine Metadaten ausgewählt worden, so beginnt das erste Kapitel des Buches automatisch auf der ersten Seite des E-Books.

Cover-Upload

Zum Hochladen einer Titelbilddatei dient das in Abb. 4-13 gezeigte Formular:

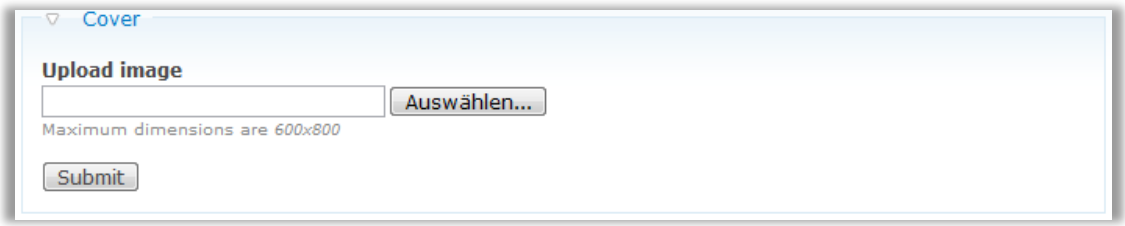

Abb. 4-13 Formular für den Cover-Upload

Über den Knopf „Auswählen“ (die Beschriftung ist abhängig vom Browser und der ausgewählten Betriebssystemsprache) kann die Bilddatei für den Upload bestimmt werden. Anzumerken ist hierbei, dass ausschließlich JPEG-, PNG- und GIF-Bilddateien

akzeptiert werden. Wurde die Datei nach dem Betätigen des Knopfes „Submit“ erfolgreich hochgeladen, erscheint eine Vorschau des Bildes, wie in Abb. 4-14 zu sehen ist.

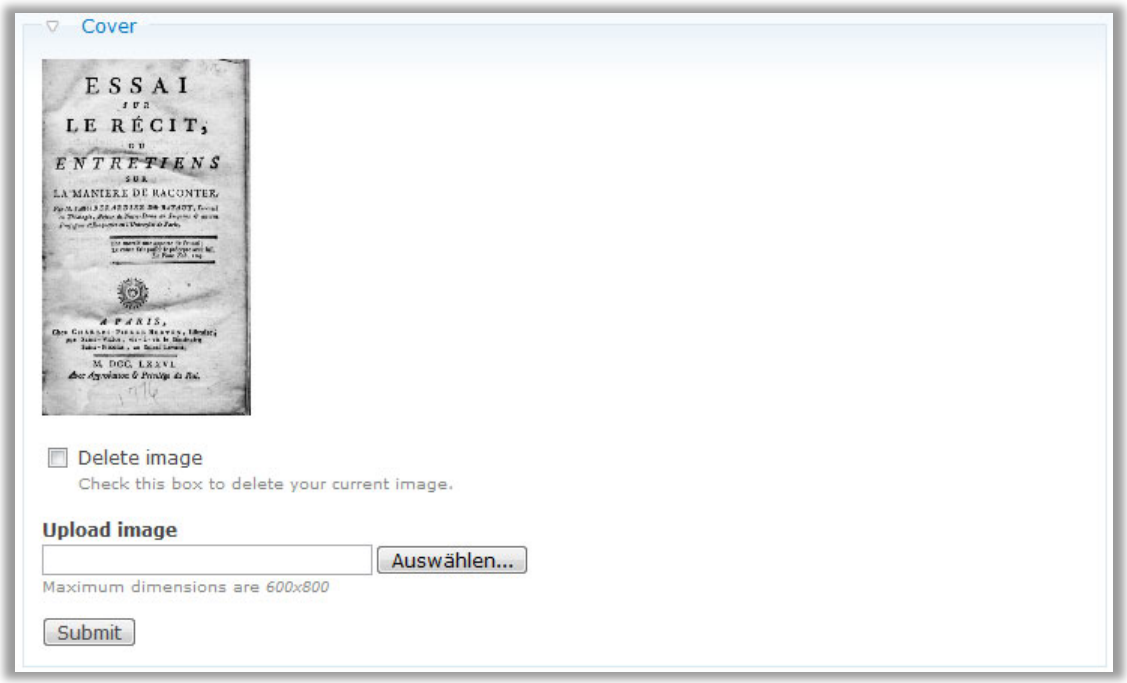

Abb. 4-14 Formular für den Cover-Upload mit Vorschau

Sollte das Bild nicht den Wünschen des Benutzers entsprechen, kann es auch wieder gelöscht werden, indem die Checkbox „Delete image“ aktiviert und der Button „Submit“ betätigt wird. Des Weiteren kann ein bereits hochgeladenes Bild durch ein anderes ersetzt werden, das vorherige wird dabei automatisch gelöscht.

4.3 Benutzung

Nach den verschiedenen Konfigurationsmaßnahmen soll nun der Umgang mit dem TEI2Pub-Block näher erläutert werden. Dieser Block dient dem Download von TEI-Dateien in unterschiedlichen Formaten und befindet sich – wie in Abb. 4-15 zu sehen ist – auf der linken Seite. Seine Position entspricht der ihm zugewiesenen Region. Der Block besitzt in der Standardeinstellung drei Icons für die EPUB-, die Plain Text- und die TEI/XML-Version. Über die ersten zwei Symbole lässt sich das aktuell angezeigte Kapitel in der korrigierten Version mit den Anmerkungen des Autors in eckigen Klammern herunterladen und über das XML-Symbol das original TEI-Dokument.

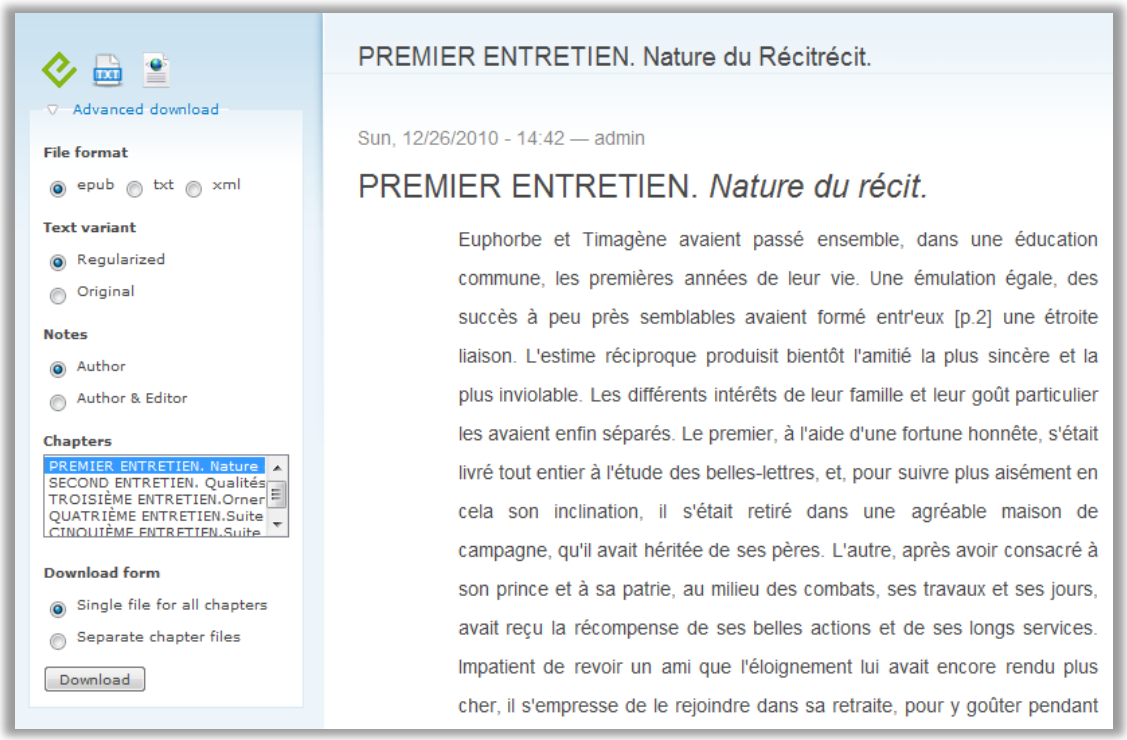

Abb. 4-15 TEI2Pub-Block mit aufgeklapptem erweiterten Download-Formular

Unter den Symbolen befindet sich der Link zum erweiterten Download. Wird dieser angeklickt, klappt sich das Download-Formular auf. Dieses Formular bietet dem Benutzer zunächst verschiedene Dateiformate für den Download, zwei verschiedene Textvarianten (die korrigierte sowie die Originalfassung) und die Wahl zwischen Anmerkungen des Autors oder denen des Autors und Herausgebers an. Des Weiteren kann der Benutzer mithilfe einer Multi-Select-Box die Kapitel des Buches auswählen, die er herunterladen möchte. In der Auswahlbox sind alle Kapitel des aktuell angezeigten Buches aufgeführt. Nun können entweder einzelne Kapitel ausgesucht oder aber durch das Drücken der Steuerungs- bzw. Shift-Taste mehrere oder sämtliche Kapitel des Buches ausgewählt werden. Zu guter Letzt besteht die Möglichkeit, die Form des Downloads zu bestimmen: In der ersten Variante gibt es eine einzelne Datei für alle ausgewählten Kapitel, in der zweiten wird für jedes gewählte Kapitel eine separate Datei erstellt und in einem ZIP-Archiv zusammengefasst.

Die folgende Abbildung zeigt die EPUB-Version des Textes „Essai sur le récit“ auf dem E-Reader. Links ist das vom TEI2Pub-Modul erstellte Inhaltsverzeichnis zu sehen, rechts das erste Kapitel:

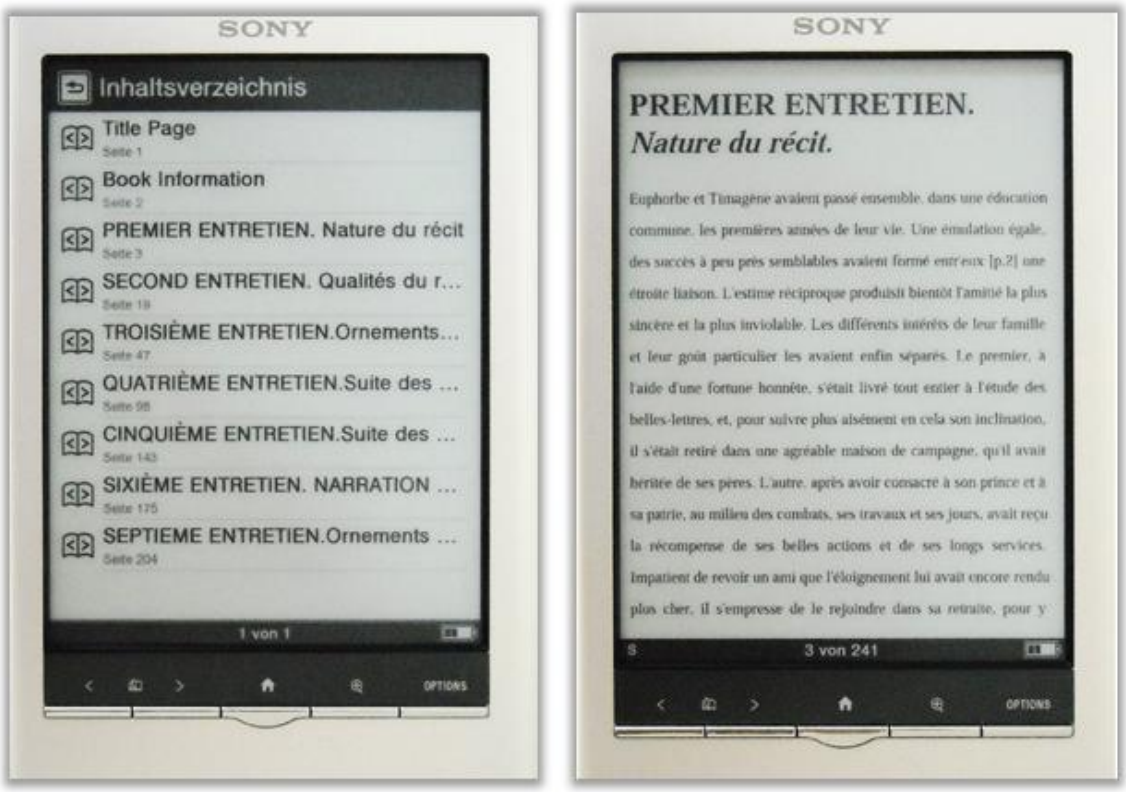

Abb. 4-16 Anzeige der EPUB-Datei

4.4 Deaktivierung und Deinstallation

Zum Deaktivieren des Moduls reicht es, den Haken in der Modul-Liste zu entfernen und auf „Speichern“ zu klicken. Das Modul ist damit deaktiviert und seine Funktionen nicht mehr verfügbar. Darüber hinaus kann das Modul nach seiner Deaktivierung auch deinstalliert werden, es entfernt dann alle Datenbankeinträge, die es im Laufe seiner Funktionalität erstellt hat.

5 Implementierung

In diesem Kapitel werden der Aufbau des TEI2Pub-Moduls und seine einzelnen Funktionalitäten näher beschrieben.

5.1 Stylesheet

Dem TEI2Pub-Modul liegt das XSL-Stylesheet zugrunde, welches in dem in Kapitel 1.1 genannten Projekt erarbeitet wurde. Dieses transformiert XML/TEI-Dokumente in HTML. Im vorliegenden Fall wird jedoch XHTML für die Erstellung des EPUB-Formats benötigt, deshalb war eine Erweiterung des ursprünglichen Stylesheets um den XML-Namensraum „xmlns=http://www.w3.org/1999/xhtml“ erforderlich.

Mithilfe des XSL-Stylesheets werden die zwei verschiedenen Textvarianten (Originalfassung und korrigierte Version) als auch die Anmerkungen zum Text (Anmerkungen des Autors bzw. Anmerkungen des Autors *und* des Herausgebers) zu CSS-Klassen gruppiert. Die nicht ausgewählte Version wird hierbei jeweils durch „display:none“ in der CSS-Datei ausgeblendet.

Die CSS-Stylesheet-Datei ist für die Darstellung der XHTML-Dateien im EPUB-Format zuständig und wurde ebenso wie das XSL-Stylesheet im erwähnten Projekt konzipiert. Einige Modifikationen waren hinsichtlich der Einstellung der Schriftgrößen (von px auf em) nötig, da die Funktion des Zoomens auf dem E-Reader mit px-Schriftgröße nicht möglich war.

Da beim Erstellen der Plain Text-Version im Gegensatz zur EPUB-Version keine CSS-Datei verwendet werden kann, um die gewünschte Textfassung ein- bzw. die nicht gewünschte auszublenden, wurden für das TEI2Pub-Modul vier weitere Stylesheets entwickelt. Diese kommen beim Transformieren von TEI-Dokumenten in die Plain Text-Version zum Einsatz. Sie entnehmen die gewünschte Textversion dem TEI-Dokument und erzeugen eine Plain Text-Version. Die vier genannten Stylesheets sind für die

- Originalfassung des Textes inklusive der Anmerkungen des Autors
- Originalfassung des Textes inklusive der Anmerkungen des Autors und Herausgebers

- korrigierte Fassung des Textes inklusive der Anmerkungen des Autors
- korrigierte Fassung des Textes inklusive der Anmerkungen des Autors und des Herausgebers

konzipiert worden.

5.2 Modulaufbau

Zunächst wird ein Verzeichnis für das in der vorliegenden Arbeit konzipierte Modul in Drupal-Installation unter „…/sites/all/modules“ angelegt. Das Verzeichnis besitzt den gleichen Namen wie das Modul, in diesem Fall „tei2pub“ und ist nach den Coding Standards von Drupal in Kleinbuchstaben geschrieben. Anschließend werden drei weitere Dateien erstellt, die das Fundament des Moduls bilden: die tei2pub.info-, tei2pub.install- und die tei2pub.module-Datei, die im Folgenden betrachtet werden.

5.2.1 tei2pub.info

Diese Informationsdatei ist obligatorisch, damit das Modul von Drupal erkannt wird. Sie enthält Meta-Informationen über das Modul wie seinen Namen, die Beschreibung der Abhängigkeiten zu anderen Modulen und die Angabe der Drupal-Version, zu der es kompatibel ist. Da Drupal beim Laden der Modul-Administrationsseite alle Module analysiert, ist eine separate „.info“-Datei nötig, damit der Speicherverbrauch gering gehalten werden kann. Das Zurückgeben von Meta-Informationen über eine Funktion des Hauptmoduls hätte möglicherweise eine Überschreitung des Speicherlimits von PHP zur Konsequenz [14, S. 46]. Nachfolgend der Inhalt der „tei2pub.info“-Datei:

```
; $Id$
name = TEI2Pub
description = "Lets you download TEI/XML files as EPUB-
Format"
core = 7.x
package = TEI2Pub
files[] = tei2pub.module
```

```
files[] = tei2pub.install
dependencies[] = teichi
configure = admin/config/tei2pub
```

Das `$Id$`-Zeichen zu Beginn der Datei ist eine CVS-Variable, deren Wert beim Hochladen des Moduls in das Module Repository von Drupal automatisch von CVS durch die Angabe der Versionsnummer, Datum, Uhrzeit und Name des Entwicklers ersetzt wird [25]. Mit `name` wird der Name des Moduls angegeben, der in der Modul-Liste im Administrationsbereich erscheint. `description` hingegen gibt die Beschreibung des Moduls wieder (siehe Abb. 4-2). `core` definiert die Drupal-Hauptversion, mit der das Modul kompatibel ist. Im vorliegenden Fall bedeutet 7.x, dass das Modul für Drupal Version 7 und alle seine Unterversionen (7.0, 7.1 usw.) geschrieben ist. Inkompatible Module lassen sich somit nicht aktivieren. Mithilfe von „Paketen“ können Module miteinander gruppiert werden, `package` gibt daher an, zu welcher Gruppe das Modul gehört. Steht beispielsweise „`package = TEI2Pub`“ bei drei verschiedenen Modulen, erscheinen diese bei der Auflistung in einer Gruppe. `dependencies` gibt an, von welchen Modulen das TEI2Pub-Modul abhängig ist [26]. Demnach kann das Modul erst dann aktiviert werden, wenn das TEIChi-Modul bereits aktiv ist. Zu guter Letzt wird der Konfigurationspfad mit `configure` bestimmt.

5.2.2 tei2pub.install

Diese Datei ermöglicht das Erstellen von Datenbanktabellen bei der Installation bzw. das Löschen bei der Deinstallation des Moduls. Für das TEI2Pub-Modul wird eine Tabelle benötigt, in welcher Bücher und die Pfade der dazugehörigen Titelbilder gespeichert und für den Download der EPUB-Version des gewünschten Buches bereitgestellt werden können.
Zunächst werden zwei Hauptfunktionen definiert: eine Install-Funktion und eine Uninstall-Funktion.

```
function tei2pub_install(){
  drupal_install_schema("tei2pub_books");
}
```

```
function tei2pub_uninstall(){
  drupal_uninstall_schema("tei2pub_books");
}
```

Wird das TEI2Pub-Modul zum ersten Mal aktiviert, sucht Drupal automatisch nach der Datei `tei2pub.install` und führt die Funktion `tei2pub_install` aus. Diese ruft ihrerseits die Funktion `drupal_install_schema` auf. Das gewünschte Schema wird hier als Funktionsparameter „`tei2pub_books`“ übergeben.

Wie bereits an anderer Stelle erwähnt, unterstützt Drupal mittels seiner Datenbankabstraktionsschicht verschiedene Typen von Datenbanken. Aus diesem Grund gibt es die Schema-API [27], welche die Allgemeingültigkeit der Module gewährleistet. Ist ein Modul beispielsweise nach MySQL-Vorgaben geschrieben worden, könnte der Fall eintreten, dass es bei der Verwendung eines anderen Datenbanktyps nicht kompatibel ist. Die Schema-API verhindert dies. Die Datenbankstruktur muss nun nur noch so formuliert werden, dass sie zur Schema-API passt und von ihr verstanden wird. Eine konkrete Transformation in datenbankverständliche Befehle (z.B. MySQL) geschieht dann automatisch. Zudem müssen die Datenbanktabellen nur einmal erstellt werden, wodurch die gleiche Beschreibung sowohl für die Installation als auch für die Deinstallation genutzt werden kann. Nachfolgend das „`tei2pub_books`“-Schema:

```
function tei2pub_books_schema(){
  $schema['tei2pub_books'] = array(
    'description' => 'Table for books',
    'fields' => array(
       'bid' => array(
         'type' => 'int',
         'not null' => TRUE,
         'default' => 0,
         'description' => ' The book ID '
         ),
      'image_path' => array(
         'type' => 'varchar',
         'length' => 32,
         'not null' => TRUE,
```

```
            'default' => '',
            'description' => Cover image path'
        ),
    ),
    'primary key' => array('bid'),
  );
  return $schema;
}
```

Mithilfe dieses Schemas wird eine Tabelle mit dem Namen `tei2pub_books` erstellt. Diese Tabelle besitzt zwei Tabellenfelder, als `bid` und `image_path` bezeichnet. Beide Felder definieren ihren Typ und zusätzliche Parameter wie ihre Beschreibung. Abbildung 5-1 zeigt das Resultat für dieses Schema als MySQL-Datenbank:

Feld	Typ	Kollation	Attribute	Null	Standard
bid	int(11)			Nein	0
image_path	varchar(32)	utf8_general_ci		Nein	

Abb. 5-1 tei2pub_books Tabelle

In Drupal können Module deaktiviert bzw. deinstalliert werden. Wird der Haken eines Moduls in der Modul-Liste entfernt und auf „Speichern“ geklickt, ist das Modul nur deaktiviert, das heißt, alle Variablen und Datenbank-Tabellen bleiben erhalten. Sollte das Modul anschließend wieder aktiviert werden, sind auch alle administrativen Einstellungen, die an ihm vorgenommen wurden, wieder verfügbar. Ist ein vollständiges Entfernen eines Moduls inklusive aller Einstellungen und Datenbankeinträge erwünscht, so muss es zuerst deaktiviert und im Anschluss über den Menüpunkt „Deinstallation“ deinstalliert werden. Für diesen Fall ist die Funktion `tei2pub_uninstall` zuständig. Diese wiederum ruft die Funktion `drupal_uninstall_schema` mit dem Funktionsparameter `tei2pub_books` auf. Die Tabelle `tei2pub_books` wird nun samt Inhalt aus der Drupal-Datenbank gelöscht.

Alle administrativen Einstellungen des Moduls werden in der Drupal-Tabelle „variable“ gespeichert. Diese Tabelle besteht aus zwei Feldern: Aus dem Feld „name“, in dem die Variablennamen aufgeführt sind und dem Feld „value“, das den Wert der Vari-

able speichert. Um die gespeicherten Einstellungen aus dieser Tabelle zu löschen, wird die Funktion `tei2pub_uninstall` um den folgenden Code erweitert:

```
$result = db_query("SELECT name FROM {variable} WHERE name
LIKE 'tei2pub%'");
while ($record = $result->fetch()) {
  variable_del($record->name);
}
```

Zuerst werden alle Variablennamen, die mit „tei2pub“ beginnen, von der Datenbank abgefragt und anschließend in einer „While“-Schleife mithilfe der von Drupal zur Verfügung gestellten Funktion `variable_del` einzeln gelöscht.

5.2.3 tei2pub.module

Die Datei „tei2pub.module“ stellt die eigentliche Funktionalität des Moduls bereit. Sie lässt sich in zwei Hauptbereiche aufgliedern: Zum einen umfasst sie die Implementierung der Funktionalitäten des administrativen Bereichs, dazu zählen die Verwaltung der Zugriffsrechte, die Textgestaltung des Download-Bereichs, die Einstellungen zur Gestaltung des E-Books als auch der Cover-Upload. Zum anderen enthält die Datei die Implementierung der Funktionalitäten des Download-Bereichs wie das Herunterladen verschiedener Dateiformate (EPUB, Plain Text, TEI/XML) und den erweiterten Download (Auswahl verschiedener Textvarianten und einzelner Kapitel eines Buches).

5.3 Funktionalitäten des administrativen Bereichs

An dieser Stelle wird die Implementierung der Funktionalitäten des administrativen Bereichs erläutert, dazu gehören die Benutzerhilfe, die Verwaltung der Zugriffsrechte, die Konfigurationsseiten und der Cover-Upload.

5.3.1 Hilfe

Das TEI2Pub-Modul stellt eine Hilfe-Funktion bzw. erweiterte Informationen bereit, die mithilfe des Hooks „help“ [28] implementiert werden. Der Name der Funktion ist „`tei2pub_help()`“:

```
function tei2pub_help($path, $arg) {
 $output = '';
  switch ($path) {
    case 'admin/help#tei2pub':
      $output = t('The TEI2Pub module allows you to down
                  load the ePUB-Version…');
      return $output;
      break;
  }
}
```

Die Variable `$path` gibt hierbei an, wo sich ein Nutzer befindet, wenn er die „tei2pub_help“-Funktion aufruft. Diese Variable wird über eine „switch“-Anweisung abgefragt. Der Fall (engl. „case“) `admin/help#tei2pub` tritt ein, wenn von der allgemeinen Drupal-Hilfeseite (/admin/help) verlinkt wird. Der Rückgabewert dieser Funktion sind Informationen zur angeforderten Hilfe in Textform. Dieser Text wird mit der Funktion `t()` für „translate“ in die bei Drupal aktivierte Sprache der Benutzeroberfläche übersetzt.

5.3.2 Verwaltung der Zugriffsrechte

An dieser Stelle soll die Realisierung der Berechtigungsfunktion erläutert werden. Die Implementierung der benutzerspezifischen Konfigurationsmöglichkeiten des TEI2Pub-Moduls für den Download-Bereich erfolgt über `hook_permission()`:

```
function tei2pub_permission() {
  return array(
    'use tei2pub' => array(
```

```
      'title'       => t('Use TEI2Pub'),
      'description' => t('Allow users to use TEI2Pub…')
    ),
  );
}
```

Diese Funktion gibt ein Array zurück, in dem der Name bzw. die Bezeichnung der Berechtigung definiert ist, im vorliegenden Fall ist es „`use tei2pub`“. Die Bezeichnung der Berechtigung ist an sich beliebig wählbar, sie muss allerdings eindeutig sein, da sie sonst durch ein bereits installiertes Modul mit der gleichen Bezeichnung überschrieben werden kann [29]. Aus diesem Grund enthält die hier gewählte Bezeichnung den Namen des Moduls, um Namenskonflikte mit anderen Modulen zu vermeiden. Mit diesem Schritt ist die Implementierung der Berechtigungsfunktion abgeschlossen. Auf der Administrationsseite von Drupal erscheint nun unter „Permissions“ ein Eintrag zu den benutzerspezifischen Richtlinien des TEI2Pub-Moduls. Die Überprüfung der Berechtigung wird mithilfe der Funktion `user_access()` ausgeführt, die in Kapitel 5.4.1 näher beschrieben wird.

5.3.3 Hook_menu

Um in dem administrativen Bereich Konfigurationsseiten für das TEI2Pub-Modul einrichten zu können, müssen zuerst Menüelemente erstellt werden, die den Administrator automatisch auf die entsprechende Konfigurationsseite weiterleiten. Diese Weiterleitung übernimmt das Menüsystem von Drupal, das unter anderem für die Zuordnung der sogenannten „Callbacks“ zuständig ist. Richtet ein Browser eine Anfrage an Drupal, übergibt er einen Pfad bzw. eine URL. Aus dieser URL leitet Drupal ab, wie die Anforderung auszuführen ist. Drupal trennt den Hostname aus der URL heraus und verwendet nur ihren zweiten Teil. Sollte die URL beispielsweise „http://hostname/admin/config“ lauten, gibt „admin/config“ den Drupal-Pfad wieder. Anhand dieses Pfads erkennt Drupal, dass seine Hauptkonfigurationsseite angefordert wird und diese dementsprechend anzeigt werden muss [14, S. 97].

Die allgemeine Vorgehensweise beim Erstellen von Menüelementen sieht folgendermaßen aus: Drupal verlangt von allen aktivierten Modulen, die Menüelemente in einem Array bereitzustellen. Jedes Menüelement besteht seinerseits aus einem Array

mit einem Pfad als Schlüssel und weiteren Informationen über den Pfad, zu diesen gehört unter anderem ein Seiten-Callback. Ein Callback ist der Name einer PHP-Funktion, die bei der Anforderung des Pfads aufgerufen wird [14, S. 98].

Der wichtigste Hook, um Menüelemente für ein Modul bereitzustellen, ist der Hook „menu“ [30]. Eine Funktion `tei2pub_menu` ist wie folgt aufgebaut:

```
function tei2pub_menu(){
  $items['admin/config/tei2pub'] = array(
    'title' => 'TEI2Pub',
    'description' => 'Configure TEI2Pub',
    'position' => 'left',
    'page callback' => 'system_admin_menu_block_page',
    'access arguments' =>
         array('access administration pages'),
    'file' => 'system.admin.inc',
    'file path' => drupal_get_path('module', 'system'),
  );

  $items['admin/config/tei2pub/settings'] = array(
    'title' => 'TEI2Pub Settings',
    'description' => 'Configure the TEI2Pub module',
    'page callback' => 'drupal_get_form',
    'page arguments' =>
      array('tei2pub_block_settings_form'),
    'access arguments' => array('administer users'),
    'weight' =>0,
  );
  […]
  Return $items;
}
```

Das erste Item besagt Folgendes: Geht der Benutzer zu „http://hostname/admin/config/tei2pub“, wird die Funktion `system_admin_menu_block_page` aufgerufen. Diese Funktion befindet sich in der Datei `system.admin.inc` von Drupal. Nur Benutzer mit der Berechtigung `ac-`

`cess administration pages` können dieses Menüelement sehen. Die Funktion `system_admin_menu_block_page` erstellt einen leeren Konfigurationsblock mit der Beschriftung TEI2Pub.

Mit dem zweiten Item wird ein Untermenü in diesem Konfigurationsblock mit der Beschriftung „TEI2Pub settings“ (vgl. Abb. 4-8) erstellt. Fordert der Benutzer nun die URL „http://hostname/admin/config/tei2pub/settings“ von Drupal an, wird die Funktion `drupal_get_form` mit der Formular-ID `tei2pub_block_settings_form` aufgerufen, welche das Konfigurationsformular für den TEI2Pub-Block verfasst.

Für das TEI2Pub-Modul wurden weitere Menüelemente erstellt, auf eine Darstellung ihres Quellcodes wurde an dieser Stelle verzichtet, da sie den gleichen Aufbau wie die oben bereits aufgeführten besitzen.

Nachdem die einzelnen Menüelemente definiert worden sind, sollen nun die dazugehörigen Konfigurationsseiten näher betrachtet werden. Wie bei der Verwendung in Kapitel 4 erwähnt, gibt es für das TEI2Pub-Modul drei verschiedene Konfigurationsseiten: eine für das Download-Formular, eine für die Konfiguration der einzelnen Bücher und schließlich eine für den Cover-Upload.

5.3.4 Konfiguration des Download-Formulars

Sämtliche Konfigurationsseiten des TEI2Pub-Moduls werden mithilfe der Form-API [31] konzipiert. Sie dient dem Erstellen, Validieren und Verarbeiten von Einstellungsformularen. Die Formulare an sich sind ein verschachteltes Array mit Eigenschaften und Werten, das beim Anzeigen der Seite mittels der Form-API in HTML transformiert wird.

In Drupal ist es unerlässlich, Formulare eindeutig zu kennzeichnen. Bei mehreren Formularen muss festgestellt werden können, welches angefordert wurde, damit es mit den Funktionen verknüpft werden kann, die für das Erstellen des Formularinhalts verantwortlich sind. Aus diesem Grund wird jedem Formular eine Formular-ID zugewiesen, die beim Aufruf von drupal_get_form() als Funktionsargument übergeben wird. Über die Formular-ID ermittelt Drupal die Namen der Validierungs-, Übermittlungs- und Theme-Funktionen. Darüber hinaus ist sie die HTML-ID im <form>-Tag, so dass ein Ändern des Formularaussehens mittels CSS-Stylesheet über diese ID möglich ist.

Die Formular-ID setzt sich in der Regel aus dem Modulnamen und einem „Bezeichner“ zusammen, der das Formular beschreibt [14, S. 285]. Im Kontext der Konfi-

guration des TEI2Pub-Download-Formulars wird die Formular-ID `tei2pub_block_settings_form` gewählt. Die Konfiguration des Download-Formulars (vgl. Abb. 4-9) wird mithilfe der nachfolgenden Funktion implementiert, die aufgrund ihres Umfangs hier nur in gekürzter Fassung dargestellt werden soll:

```
function tei2pub_block_settings_form(){
  $form = array();
  $form['advanced_download_link'] = array(
    '#type' => 'fieldset',
    '#title' => t('Advanced download link'),
    '#collapsible' => TRUE,
    '#collapsed' => TRUE,);
  $form['adv…']['tei2pub_advanced_download_link'] = array(
    '#type' => 'textfield',
    '#title' => t('Label'),
    '#description' => t('Text for the advanced download
     link'),
    '#default_value'=> variable_get(
    'tei2pub_advanced_download_link','Advanced download'),
    '#size' => 30,);
  […]
  Return system_settings_form($form);
}
```

Das Formular wird in verschiedene Feldgruppen (engl. „fieldsets“) unterteilt, damit eine bessere Übersicht gewährleistet ist. Jedes Formularattribut wird mit einer Raute (#) gekennzeichnet, die als Array-Schlüssel fungiert. Der Typ des ersten Formularelements wird über das Attribut `#type` als `fieldset` definiert. Über das Attribut `#collapsible` wird Drupal aufgefordert, die Feldgruppe durch einen Klick auf ihren Titel auf- bzw. zuzuklappen. Standardmäßig ist die Feldgruppe zugeklappt, da das Attribut `#collapsed` auf `TRUE` gesetzt ist. In dieser Feldgruppe wird mithilfe des zweiten Formularelements ein Textfeld der Länge 30 erzeugt (vgl. Abb. 4-9). Sein Standardwert wird über `#default_value` definiert. Drupal ermöglicht es, jeden Wert mit der Funktion `variable_get()` zu speichern und abzurufen. Diese Werte werden in der Datenbank in der Tabelle „variable“ gespeichert und sind stets über eine Datenbankab-

frage verfügbar. Die Funktion `variable_get()` wird verwendet, um administrative Einstellungen ohne großen Arbeitsaufwand zu speichern, da sonst ein zusätzliches Erstellen einer Tabelle zum Speichern dieser Einstellungen notwendig wäre. Die Funktion `variable_get()` wird immer mit zwei Parametern aufgerufen: Der erste Parameter ist der Name der Variable, der zweite ist ein Standardwert, der zurückgegeben wird, falls die Variable nicht existiert. Sollte beim Aufruf der Funktion

```
variable_get('tei2pub_advanced_download_link','Advanced
download')
```

die Variable `tei2pub_advanced_download_link` in der Datenbank also nicht existieren, so wird „`Advanced download`“ zurückgegeben.

Der Name der Variablen beginnt immer mit „tei2pub“, wodurch beim Deinstallieren des TEI2Pub-Moduls alle dazugehörigen Einstellungsvariablen erkannt und aus der `variable`-Tabelle gelöscht werden können.

Mithilfe der Funktion `system_settings_form()` wird für das erstellte Formular ein „Speichern“-Knopf erzeugt, die Abarbeitung des Formulars inklusive des Speicherns der vorgenommenen administrativen Einstellungen erfolgt automatisch.

5.3.5 Konfiguration der Bücher

Zuerst werden alle verfügbaren Bücher in Drupal mithilfe der Funktion `_tei2pub_get_books()` abgerufen. Diese Funktion erstellt eine Datenbankabfrage und verlangt alle Bücher, die das Body-Format „teichinode“ haben, also TEI-Dokumente sind. Das Ergebnis ist ein Array, dessen Schlüssel die Buch-ID und dessen Wert der Titel des Buches ist. In einer „for“-Schleife werden für jedes Buch zwei Hyperlinks erzeugt. Für die Konfiguration des Buches ist dies der folgende Hyperlink:

```
<a href="url(admin/config/tei2pub/books_settings/edit/)".
$bid > Configure settings </a>
```

Für den Cover-Upload sieht der Hyperlink wie folgt aus:

```
<a href="url(admin/config/tei2pub/books_settings/cover/)".
$bid> Cover upload </a>.
```

„$bid“ gibt hierbei die Buch-ID wieder. Diese Hyperlinks werden in einem Array gespeichert und schließlich der von Drupal zur Verfügung gestellten Funktion `theme_table()` übergeben. Diese Funktion dient der tabellarischen Darstellung aller Bücher und der dazugehörigen Aktionen („Configure settings“ bzw. „Cover upload“, vgl. Abb. 4-10). Die Pfade in den Hyperlinks sind Menüelemente, die bereits mithilfe des Hooks „menu“ definiert wurden. Beim Anklicken des „Configure settings“-Links wird die Callback-Funktion `tei2pub_book_settings_form` aufgerufen. Anhand der Book-ID kann festgestellt werden, um welches Buch es sich handelt und welche Konfigurationsseite nun angezeigt werden muss.

Die Funktion `tei2pub_book_settings_form` erstellt das in Abb. 4-11 zu sehende Formular für die Konfiguration der einzelnen Bücher. Auch hier werden die administrativen Einstellungen mittels der Funktion `variable_get()` in der „variable“-Tabelle gespeichert.

5.3.6 Konfiguration des Cover-Uploads

Beim Anklicken des „Cover upload“-Links wird die Callback-Funktion `tei2pub_book_cover_upload_form` aufgerufen. Diese erzeugt das Formular, das dem Cover-Upload dient. Es ist in zwei Bereiche aufgeteilt, in einen Bereich für die Vorschau und einen für den eigentlichen Upload. Zuerst wird jedoch für die übergebene Book-ID geprüft, ob zu dem Buch bereits ein Bild hochgeladen wurde:

```
db_query("SELECT image_path FROM {tei2pub_books} t WHERE
t.bid = $bid");
```

Ist dies der Fall, wird ein Array mit der Bezeichnung „variables“ erstellt, das Informationen wie den Pfad zu dem Bild sowie die Höhe und Breite der Vorschau enthält:

```
$variables = array(
  'path' => $image_path,
  'width' => '30%',
  'height' => '30%',
);
```

Das Array mit den oben genannten Eigenschaften wird der von Drupal zur Verfügung gestellten Funktion `theme_image` übergeben. Diese erzeugt ein <img>-Tag, das für die Darstellung des Bildes zuständig ist.

```
$form['cover_setting']['image'] = array(
  '#markup' => theme('image', $variables),
);
```

Das Formularelement für das Upload-Feld wird wie folgt implementiert:

```
$form['cover_setting']['image_upload'] = array(
  '#type' => 'file',
  '#title' => t('Upload image'),
  '#size' => 30,
);
```

`'#type' => 'file'` besagt, dass es sich hierbei um einen File-Upload handelt, der ein Feld für die Pfadeingabe als auch einen Auswahlknopf besitzt, welcher seinerseits einen Dateiauswahldialog beim Betätigen bereitstellt. Am Ende des Formulars wird ein Submit-Button platziert, der die Funktion
`tei2pub_book_cover_upload_form_submit`
beim Anklicken aufruft. Diese Funktion prüft zunächst die Nutzereingaben auf Richtigkeit, das heißt, es wird kontrolliert, ob die übergebene Datei zulässig ist. Die Überprüfung geschieht über die Definition einer Validator-Variablen, die die Dateiendung überprüft. Zugelassen sind ausschließlich Bilder des Dateityps JPEG, PNG und GIF. Sollte die Dateiendung nicht akzeptiert werden, wird der Benutzer mittels einer Fehlermeldung darüber informiert. Ist die Dateiendung hingegen valide, so wird ein Ordner – falls dieser nicht schon existiert – mit der Bezeichnung „TEI2Pub“ in der Drupal-Installation unter „…`\sites\default\files`“ angelegt, das Titelbild dort abge-

speichert und ein Eintrag in der Datenbanktabelle `tei2pub_books` vorgenommen. Letzterer beinhaltet die Buch-ID und den dazugehörigen Pfad des Bildes. Schließlich wird der Status der hochgeladenen Datei auf „FILE_STATUS_PERMANENT“ gesetzt, da Drupal die Datei sonst als temporär klassifiziert und nach einigen Stunden automatisch löscht.

Es sei darauf hingewiesen, dass der oben genannte Ordner Lese- und Schreibrechte besitzt, um die Bilder speichern zu können.

5.4 Funktionalitäten des Download-Bereichs

Im Folgenden soll die Implementierung der Funktionalitäten des Download-Bereichs expliziert werden, hierzu zählen der Download-Block, der einfache sowie der erweiterte Download.

5.4.1 Download-Block

Da der Download-Bereich als Block realisiert ist, wird zunächst der Hook „block_info“ [32] implementiert, um den Block zu definieren.

```
function tei2pub_block_info() {
  $block['tei2pub'] = array (
    'info' => t('TEI2Pub'),
    'status' => TRUE,
    'region' => 'Left',
    'cache' => DRUPAL_CACHE_GLOBAL,
    'weight' => 0,
  );
  return $block;
}
```

Der „info“-Wert ist erforderlich, da er eine Beschreibung des Blocks enthält, die auf der Administrationsseite der Blöcke angezeigt wird. Über „status => TRUE“ wird der Block standardmäßig beim Installieren des TEI2Pub-Moduls aktiviert.

Über „region“ wird seine Standardregion festgelegt, dennoch kann ihm auch eine neue Region auf der Block-Administrationsseite zugewiesen werden. Ist die Region in dem aktuell benutzten Theme nicht definiert, wird der Block deaktiviert.

Mit „cache“ lässt sich die Art der Zwischenspeicher bestimmen, mögliche Werte sind: DRUPAL_NO_CACHE (den Block nicht zwischenspeichern), DRUPAL_CACHE_PER_ROLE (für die Rolle des Benutzers zwischenspeichern), DRUPAL_CACHE_PER_USER (für jeden Benutzer zwischenspeichern), DRUPAL_CACHE_PER_PAGE (für jede Seite zwischenspeichern) und DRUPAL_CACHE_GLOBAL (den Block nur einmal für alle Benutzer zwischenspeichern). Um den Aufruf der Seiten zu beschleunigen und das Netz zu entlasten, wurde im vorliegenden Fall die letzte Möglichkeit gewählt [14].

„weight“ dient der Anordnung eines Blocks nach Gewichtung. Sollten mehrere Blöcke in einer bestimmten Region angezeigt werden, so steigt ein Block mit einem geringeren Gewicht in der Region vertikal nach oben oder horizontal nach links. Besitzt der Block ein höheres Gewicht, sinkt er nach unten oder auf die rechte Seite der Region.

Nachdem der Block definiert ist, wird sein Inhalt mithilfe des Hooks „block_view“ implementiert. Zuerst wird die aktuell angezeigte Node-ID aus der URL ermittelt, anschließend erfolgt ein Aufruf der vom TEIChi-Modul implementierten Funktion `teichi_get_teichi_nodes`. Diese Funktion gibt eine Liste aller TEIChi-Nodes zurück. Anhand dieser Liste wird nun überprüft, ob der angezeigt Node ein vom TEIChi-Modul visualisiertes TEI-Dokument ist. Ist dies der Fall, werden die Zugriffsrechte des Benutzers einer Kontrolle unterzogen. Sie werden mithilfe der Funktion `user_access` geprüft. Hat der Benutzer ausreichende Berechtigungen, wird das Download-Formular angezeigt.

```
function tei2pub_block_view($delta = '') {
  $nid = arg(1);
  $teichi_nodes = teichi_get_teichi_nodes();
  foreach($teichi_nodes as $line) {
    $teichi_nid = $line->nid;
    if($nid == $teichi_nid && user_access('use tei2pub')){
$block['content']=drupal_get_form("tei2pub_download_form");
      return $block;
    }
  }
```

```
}
```

Das Download-Formular wird von der Funktion `drupal_get_form ("tei2pub_download_form")` erzeugt. Diese prüft zunächst die für den Download aktivierten Dateiformate. Für jedes Dateiformat wird ein Knopf wie nachfolgend zu sehen, implementiert:

```
$form['epub'] = array(
  '#type' => 'submit',
  '#name' => 'epub_download_button',
    '#button_type' => 'image',
    '#attributes' => array(
      'class' => array('tei2pub_Button'),
      'id' => array('tei2pub_epub_Button')),
    '#weight' => 0,
);
```

Über „type“ wird das Formularelement als Knopf definiert, mit „name“ wird diesem ein eindeutiger Name gegeben. Anhand dieses Namens wird später der angeklickte Knopf identifiziert und dementsprechend das gewünschte Dateiformat heruntergeladen. Drei Knöpfe wurden implementiert: `epub_download_button`, `txt_download_button` und `xml_download_button`.

Über `'#button_type' => 'image'` wird der Typ des Knopfes festgelegt. „image“ bedeutet, dass es sich hierbei um einen grafischen Button handelt, das heißt, ein Bild (Icon) repräsentiert den Knopf. Die Zuweisung des Icons zu den drei verschiedenen Knöpfen geschieht über eine CSS-Datei. Aus diesem Grund werden für jeden Button eine Klasse sowie eine ID definiert, die ihn eindeutig identifizierbar macht. Alle drei Buttons gehören der gleichen Klasse an.

Über diese Klasse werden die Position, der Cursor-Typ, die Breite, die Höhe und der Rahmen des Icons bestimmt. Die Breite und die Höhe wurden auf 26px gesetzt. Fährt der Benutzer mit der Maus über die Icons, nimmt der Cursor die Gestalt eines Zeigers an. Nachfolgend ist der CSS-Abschnitt hierfür zu sehen:

```
.tei2pub_Button {
   margin: 0px 10px 0px 0px;
   cursor: pointer;
   width:26px;
   height:26px;
   border:none;
}
```

Über die ID wird jedem Knopf sein Icon zugewiesen. Die Icons befinden sich, wie schon an anderer Stelle erwähnt, in einem Ordner mit der Bezeichnung „images“. Es folgt der CSS-Abschnitt dazu:

```
#tei2pub_epub_Button {
    background: url(../images/epub.png);
}
```

Zudem enthält das Formular weitere Optionen für den erweiterten Download. Jede Option wird als ein Formularelement mit einem eindeutigen Namen implementiert. Dieser dient später der Auswertung des Formulars „Erweiterter Download“.

5.4.2 Einfacher Download

Beim Anklicken einer der drei Icon-Knöpfe wird die Funktion tei2pub_download_form_submit() aufgerufen. Diese bekommt eine Variable mit der Bezeichnung „form_state“ übergeben, die alle Informationen über das aufrufende Formular enthält.

Zuerst wird mit der folgenden Code-Zeile der Namen des angeklickten Icon-Knopfes ermittelt:

```
$button = $form_state['clicked_button']['#name'];
```

Anschließend wird der Name in einer „switch-case“-Anweisung überprüft und die entsprechende Funktion aufgerufen:

```
switch ($button) {
  case 'epub_download_button':
    _tei2pub_download_chapter_as_epub($nid);
    break;
  case 'txt_download_button':
    _tei2pub_download_chapter_as_txt($nid);
    break;
  case 'xml_download_button':
     _tei2pub_download_chapter_as_xml($nid);
     break;
}
```

EPUB-Version

Wird die EPUB-Version verlangt, so erfolgt der Aufruf der Funktion `_tei2pub_download_chapter_as_epub($nid)`, wobei „`$nid`“ die Node-ID der aktuell angezeigten Seite wiedergibt. Diese Funktion ruft eine weitere Funktion auf, die das TEI-Dokument in ein EPUB-Format transformiert. Hierfür wird zunächst ein ZIP-Archiv mit dem Befehl `$zip = new ZipArchive()` erzeugt. Dann folgt das Hinzufügen der mimetype-Datei, die den Lesegeräten vermittelt, dass es sich um eine EPUB-Datei im ZIP-Format handelt. Der Befehl lautet: `$zip->addFromString('mimetype', 'application/epub+zip')`. Die `AddFromString`-Funktion erwartet den Namen der mimetype-Datei als erstes Argument sowie den Inhalt der Datei als zweites Argument. Auf diese Weise wird die Struktur der EPUB-Datei erstellt (vgl. Kapitel 2-3).

Als nächstes wird überprüft, ob auch ein Herunterladen des Titelbildes auf der Konfigurationsseite des Buches aktiviert ist. Sollte dies der Fall sein, wird der Pfad des Bildes mit einer Datenbank-Abfrage ermittelt und anschließend in das ZIP-Archiv in den Ordner OPS mit dem folgenden Befehl hinzugefügt:

```
$zip->addFile($cover_path,'OPS/' . $cover_name)
```

Dann wird die Funktion `_tei2pub_get_first_page($cover_name)` aufgerufen, die einen XHTML-Code zum Referenzieren des Bildes erstellt:

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
  <body>
    <div style="text-align: center;">
      <img src="$cover_name" style="width:100%;"/>
    </div>
  </body>
</html>
```

Dieser Code wird in der Datei „title.xhtml“ im OPS-Ordner gespeichert. Nun wird ein Manifest-Item der OPF-Datei und ein navPoint-Item für das Inhaltsverzeichnis der NCX-Datei hinzugefügt. Falls die Angabe von Metadaten auf der zweiten Seite des E-Books erwünscht ist, besteht der nächste Schritt im Erstellen einer weiteren XHTML-Datei, die diese umfasst. Zu diesem Zweck wird die Funktion `_tei2pub_get_second_page()` aufgerufen. Diese Funktion entnimmt alle Metadaten aus dem Header des TEI-Dokuments und gibt diese als XHTML-Code zurück. Wie bei der Titelseite werden auch hier ein Manifest-Item und ein Navigationselement erzeugt.
Die Umwandlung der TEI/XML-Datei in XHTML erfolgt als nächstes. Hierfür wird der Inhalt des TEI-Dokuments in einem Document Object Model-, kurz DOM-Element mit dem Befehl `$dom->loadXML($tei_doc)` gespeichert. Über einen XSL-Prozessor und das XSL-Stylesheet wird das DOM-Element in XHTML transformiert.

```
$proc = new XsltProcessor();
$proc->importStylesheet($xsl);
$xhtml = $proc->transformToXML($dom);
```

Das Ergebnis wird in einer Datei mit der Bezeichnung „chapter.xhtml“ im OPS-Ordner gespeichert. Zusätzlich werden auch hier ein Manifest-Item und ein Navigationselement hinzugefügt. Das ZIP-Archiv steht nun zum Download bereit.

Das Senden des ZIP-Archivs an den Benutzer übernimmt die Header-Funktion von PHP. Für das Übertragen des Archivs stehen `Content-Type` und `Content-Disposition` zur Verfügung:

```
header('Content-Type: application/epub')
header("Content-Disposition: attachment;
       filename=\"" . $filename . ".epub\"")
```

Über `Content-Type` wird der Anwendungstyp festgelegt, der für das Öffnen dieser Datei zuständig ist. `Content-Disposition` gibt dagegen den Dateinamen als auch die Dateierweiterung („epub“) wieder. Hierbei fungiert der Kapitelname als Name der Datei, der prinzipiell zwischen Anführungszeichen stehen sollte. Dies ist wichtig, damit seine korrekte Anzeige auch bei Kapitelnamen, die Leerzeichen beinhalten, gewährleistet ist.
Mit `readfile($zip)` wird schließlich das ZIP-Archiv angegeben, das heruntergeladen wird.

Das Ergebnis der Header-Funktion soll durch die folgende Abbildung veranschaulicht werden:

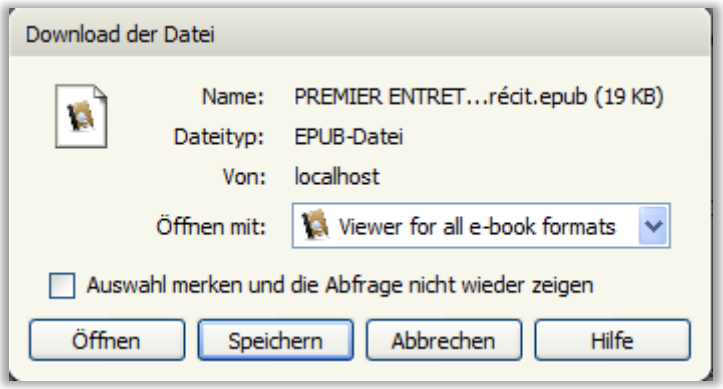

Abb. 5-2 Download-Dialog

Plain Text-Version

Bei der Plain Text-Version wird zuerst das XSL-Stylesheet bestimmt, das eine korrigierte Fassung des Textes mit Anmerkungen des Autors erzeugt:

```
$xsl=drupal_get_path('module','tei2pub').'/xsl/text_corr_reg_author.xsl'
```

Über einen XSL-Prozessor wird das TEI-Dokument mithilfe des XSL-Stylesheets in Plain Text transformiert. Das Ergebnis wird dann über die Header-Funktion zum Download angeboten.

XML-Version

Bei der XML-Version wird das TEI-Dokument direkt mithilfe der Header-Funktion zum Download angeboten, eine Transformation ist hierbei nicht nötig.

5.4.3 Erweiterter Download

Beim Anklicken des Buttons „Download“ im erweiterten Formular wird die Funktion `tei2pub_download_form_validate()` aufgerufen. Diese überprüft, ob Kapitel in der Auswahlliste markiert worden sind. Ist dies nicht der Fall, erscheint eine Fehlermeldung:

```
$button = $form_state['clicked_button']['#value'];
$chapters = $form_state['values']['chapter_list'];
if($button == "Download" && count($chapters)==0){
   form_set_error(t('Chapters list is required.'));
}
```

Sind Kapitel hingegen ausgewählt worden, werden diese in einer „for“-Schleife in das gewünschte Format umgewandelt. Die einzelnen Schritte dieser Transformation sind mit denen des einfachen Downloads (vgl. Kapitel 5.4.2) größtenteils identisch.

Die vom Benutzer ausgewählten Optionen (Dateiformat, Textvariante usw.) werden in einer „form_state“-Variablen gespeichert. Diese wird anschließend ausgewertet und der jeweils entsprechenden Funktion übergeben, die schließlich das richtige Dateiformat mit den gewünschten Optionen erstellt.

6 Fazit

Zusammenfassend lässt sich feststellen, dass die Zielsetzung dieser Diplomarbeit erfolgreich realisiert werden konnte. Mit dem entwickelten TEI2Pub-Modul können TEI-Dokumente in das EPUB- sowie in das Plain Text-Format konvertiert und zum Download angeboten werden. Das TEI2Pub-Modul zeichnet sich hierbei durch seine einfache und komfortable Bedienbarkeit aus: Drei Icons für das jeweilige Dateiformat ermöglichen den schnellen Download des aktuell angezeigten Buchkapitels in der korrigierten Textfassung inklusive der Anmerkungen des Autors.

Der erweiterte Download bietet dagegen verschiedene Einstellungsmöglichkeiten bezüglich des Dateiformats für den Download, der Textvariante, der Anmerkungen zum Text und der Kapitelauswahl. Zudem kann der Benutzer die Form des Downloads – eine zusammenhängende Datei oder aber separate Dateien in einem ZIP-Archiv – bestimmen.

Die benutzerfreundliche Konzeption des TEI2Pub-Moduls zeigt sich auch hinsichtlich seiner Konfigurationsmöglichkeiten. Es wurde so implementiert, dass es nicht sprachgebunden ist, das heißt, die Beschriftungen des Download-Formulars als auch der Text für die Dateiversion im EPUB-Format können jeweils den individuellen Wünschen angepasst werden.

Darüber hinaus ist es möglich, die einzelnen Bücher unabhängig voneinander zu konfigurieren, wodurch ein hohes Maß an Flexibilität gewährleistet ist: Für jedes Buch kann eine individuelle Einstellung in Bezug auf die Dateiformate zum Download, die Gestaltung der Titelseite und die Anzeige der Metadaten vorgenommen werden.

Für die Implementierung des Moduls fanden sich ausreichend Informationen zur EPUB-Struktur als auch zur Entwicklung eines Drupal-Moduls, einige Schwierigkeiten zeigten sich jedoch bei der Umsetzung des Cover-Uploads. Zu diesem Thema existierten nur wenige Informationsquellen, da sich Drupal 7 zum Zeitpunkt der Realisierung des TEI2Pub-Moduls noch in der Entwicklungsphase befand.

Allgemein betrachtet verlief die Implementierung des Moduls dennoch unproblematisch. Während der Anfertigung dieser Diplomarbeit wurden zahlreiche Modifikationen vorgenommen, die der Optimierung seiner Benutzerfreundlichkeit dienten. Bis zuletzt ergaben sich aus der Kooperation mit Herrn Dr. des. Christof Schöch diesbezüglich neue Ideen und Verbesserungsvorschläge. Die aktuelle Version des TEI2Pub-Moduls kommt bereits auf http://www.berardier.org/essai zum Einsatz.

Für die Zukunft zeichnen sich verschiedene Erweiterungsmöglichkeiten des TEI2Pub-Moduls ab. So kann es zum Beispiel um das Angebot anderer Dateiformate für den Download wie PDF und OpenDocument ergänzt werden. Ferner ist auch eine Erweiterung des XSL-Stylesheets möglich, so dass alle Elemente des TEI-Lite-Standards berücksichtigt werden könnten.

Ein letztes Ziel besteht schließlich darin, das TEI2Pub-Modul der Drupal-Community zur Verfügung zu stellen, um dem Verständnis Drupals als „Social Software“ Rechnung zu tragen.

Literaturverzeichnis

[1] Christof Schöch. [Zugriff am: 03. 12 2010.] http://www.christof-schoech.de/essai-sur-le-recit.

[2] Wikipedia. E-Book. [Zugriff am: 15. 01 2011.] http://de.wikipedia.org/wiki/E-Book.

[3] Text Encoding Initiative. [Zugriff am: 10. 12 2010.] http://www.tei-c.org/index.xml.

[4] TEI in der Praxis. [Zugriff am: 10. 12 2010.] http://computerphilologie.uni-muenchen.de/praxis/teiprax.html.

[5] Wikipedia. Text Encoding Initiative. [Zugriff am: 10. 12 2010.] http://de.wikipedia.org/wiki/Text_Encoding_Initiative.

[6] Borghoff, Uwe M. et al. *Langzeitarchivierung. Methoden zur Erhaltung digitaler Dokumente.* Heidelberg : dpunkt.verlag, 2003.

[7] Wikipedia. [Zugriff am: 19. 12 2010.] http://de.wikipedia.org/wiki/Drupal; Zugriff am 19.12.2010.

[8] Graf, Hagen. *Drupal 6: Websites entwickeln und verwalten mit dem Open Source-CMS.* München : Addison-Wesley, 2008.

[9] Joomla! [Zugriff am: 19. 12 2010.] http://www.joomla.de/.

[10] TYPO3.[Zugriff am: 19. 12 2010.] http://typo3.org/.

[11] Drupalbasic. [Zugriff am: 20. 12 2010.] http://drupalbasic.de/book/export/html/38.

[12] Drupalcenter. [Zugriff am: 21. 12 2010.] http://www.drupalcenter.de/handbuch/4437.

[13] Drupal. [Zugriff am: 20. 12 2010.] http://drupal.org/.

[14] VanDyk, John K. *Das Drupal-Entwicklerhandbuch: Der Praxisleitfaden für Drupal-basierte Webprojekte.* München : Addison-Wesley, 2009.

[15] Drupal API. [Zugriff am: 20. 12 2010.] http://api.drupal.org/api/drupal/includes--module.inc/group/hooks/7.

[16] International Digital Publishing Forum. [Zugriff am: 10. 12 2010.] http://www.idpf.org/.

[17] Schneider, Sara. *E-Book-Markt 2009: Analyse und Entwicklung des E-Book-Marktes im deutschprachigen Raum.* Hamburg : Diplomica Verlag, 2010.

[18] Wikipedia. EPUB. [Zugriff am: 27. 11 2010.]

http://de.wikipedia.org/wiki/EPUB.

[19] Data2type. EPUB. [Zugriff am: 27. 11 2010.]
http://www.data2type.de/Vortraege/ePub/ePub.pdf.

[20] Pagina. Gesamtherstellung Wissenschaftlicher Werke. [Zugriff am: 27. 11 2010.]
http://www.pagina-online.de/index.php?id=312&L=o&type=1 .

[21] epub eBooks Tutorial. [Zugriff am: 28. 11 2011.]
http://www.jedisaber.com/ebooks/tutorial.asp.

[22] Boeh Schaf. [Zugriff am: 28. 11 2010.]
http://www.boeh-schaf.de/Internet/epub.php.

[23] Redding, Jacob . *Jacob Redding: Beginning Drupal. Indianapolis 2010.*
Indianapolis, Indiana : Wiley Publishing, Inc., 2010.

[24] Coding standards. [Zugriff am: 26. 12 2010.]
http://drupal.org/coding-standards.

[25] Das erste eigene Modul. [Zugriff am: 11. 01 2011.]
http://www.dvfux.de/book/export/html/9.

[26] Writing .info files (Drupal 7.x). [Zugriff am: 11. 01 2011.]
http://drupal.org/node/542202.

[27] Schema API. [Zugriff am: 12. 01 2011.]
http://api.drupal.org/api/drupal/includes--database.inc/group/schemaapi/6.

[28] hook_help. [Zugriff am: 15. 01 2011.]
http://api.drupal.org/api/drupal/developer--hooks--core.php/function/hook_help/6.

[29] Erstellen von Modulen - ein Handbuch: Drupal 6.x. [Zugriff am: 17. 01 2011.]
http://www.drupalcenter.de/handbuch/15755.

[30] hook_menu. [Zugriff am: 20. 01 2011.]
http://api.drupal.org/api/drupal/developer--hooks--core.php/function/hook_menu/6.

[31] Form-API. [Zugriff am: 20. 01 2011.]
http://drupal.org/node/751826.

[32] hook_block_info. [Zugriff am: 01. 21 2011.]
http://api.drupal.org/api/drupal/modules--block-block. api.php /function/
hook_block_info/7.

[33] Wikipedia. Drag_and_Drop. [Zugriff am: 31. 12 2010.]
http://de.wikipedia.org/wiki/Drag_and_Drop.