media2mult – Ein Wiki-basiertes Autorenwerkzeug zur kollaborativen Erstellung multimedialer Dokumente

Martin Gieseking, Oliver Vornberger
Universität Osnabrück

Abstract: Bei *media2mult* handelt es sich um ein Plug-In für PmWiki, das zum einen Erweiterungen zum Einbinden unterschiedlichster Mediendateien und Skriptsprachen in Wikiseiten implementiert, und darüber hinaus eine Cross-Media-Publishing-Komponente bereitstellt, mit deren Hilfe einzelne Wikiseiten oder beliebige Seitensequenzen in verschiedene Dateiformate, wie PDF oder RTF konvertiert werden können. Dieser Beitrag soll einen Überblick über die Erweiterungen, deren Funktionsweise und Implementationskonzepte von media2mult geben.

1 Einleitung

Spätestens seit der Gründung der freien Web-Enzyklopädie Wikipedia und ihres zunehmenden Popularitätsgewinns gehören die von Ward Cunningham eingeführten Bezeichnungen *WikiWikiWeb*, *Wiki-Web* oder schlicht *Wiki* zu den nahezu allgemein bekannten Begriffen im Umfeld des Web 2.0, wenn auch deren genaue Bedeutung bzw. die sich dahinter verbergenden Konzepte zunächst oft unklar bleiben. Die zentrale Funktionalität jedes Wiki-Systems besteht darin, Webseiten mit Hilfe einer leicht erlernbaren Markup-Sprache direkt im Browser bearbeiten zu können. Damit entfällt für den Anwender die Einarbeitung in HTML und insbesondere das Hochladen der Dateien auf den Webserver. Ohne die Zuhilfenahme zusätzlicher Software lassen sich Inhalte sehr schnell online erstellen und nachbearbeiten (vgl. [LC01], S. 14–24 und [TG05], S. 164).

Das browser- und server-basierte Konzept führt dazu, dass es mehreren Autoren ermöglicht wird, orts- und zeitunabhängig an gemeinsamen Dokumenten zu arbeiten, ohne die jeweils aktuellen Fassungen untereinander austauschen zu müssen, denn der aktuelle Stand befindet sich immer auf dem Server. Die in aller Regel vorgenommene automatische Versionierung sorgt dafür, dass jederzeit zu früheren Versionen zurückgesprungen werden kann und dadurch versehentliche oder mutmaßliche Änderungen am Dokument quasi ausgeschlossen sind.

Im Zusammenhang mit der kollaborativen Erstellung von Dokumenten stellte sich an unserer Hochschule nun die Frage, ob Möglichkeiten existieren, mit Hilfe eines Wiki-Systems wissenschaftliche Texte inklusive Formeln, Grafiken und Fußnoten zu erstellen, und diese aus dem Wiki heraus in PDF-Dateien zu konvertieren. Das bisher erfolgreich eingesetzte Cross-Media-Publishing-Autorenwerkzeug *mas2tex* (vgl. [HTVW] und [HTV99]), welches ebenfalls an unserer Hochschule entwickelt wurde, konnte dies nicht leisten.

Als Resultat der Arbeit an dieser Fragestellung ist die umfangreiche PmWiki-Erweiterung *media2mult* entstanden, die heute in allen Feldern unserer zentralen Wikifarmen aktiviert

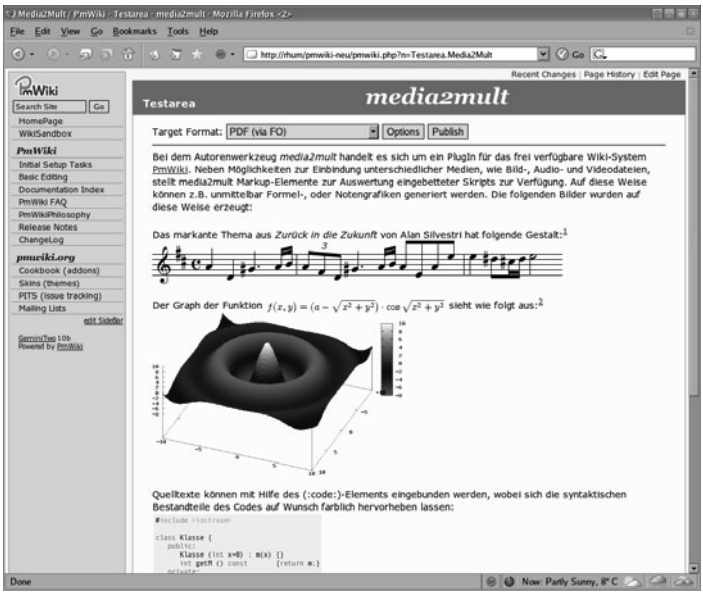

Abbildung 1: Screenshot einer Wikiseite mit verschiedenen von media2mult realisierten Dokumentkomponenten (LaTeX-Formeln, Fußnoten, Notengrafik, gnuplot-Graphen, kolorierte Quelltexte).

ist und inzwischen in über 150 Projekten zum Einsatz kommt. Zum einen nutzen Dozenten der Universität und Fachhochschule Osnabrück die erweiterten Wikis zur Aufbereitung von Vorlesungsmaterialien und zum anderen werden in einigen Fachbereichen studentische Seminararbeiten mit media2mult verfasst. Abbildung 1 zeigt eine beispielhafte Wikiseite, die mit Hilfe von media2mult durch verschiedene Content-Bestandteile angereichert wurde.

2 PmWiki

Da der Begriff *Wiki-Web* lediglich ein Konzept, aber keinen umfassenden Standard beschreibt, existiert heute eine Vielzahl unterschiedlichster Wiki-Implementationen. Dabei reicht das Spektrum von einfachen, minimalistischen Ansätzen bis hin zu komplexen und erweiterbaren Systemen. Ebenso vielfältig sind auch die Varianten der eingesetzten Eingabesprachen, sowohl was die Syntax als auch den generellen Auszeichnungsvorrat betrifft.[1]

Zu den umfangreicheren Wiki-Varianten gehört u.a. das freie, von Patrick Michaud in PHP entwickelte PmWiki[2]. Es zeichnet sich neben der einfachen Installation insbesondere durch seine hohe Konfigurierbarkeit und die nahezu unbegrenzte Erweiterbarkeit aus. Diesen Eigenschaften ist es zu verdanken, dass sich mittlerweile eine relativ große Entwicklergemeinde gebildet hat, die verschiedene Lösungsansätze zu diskutierten Problem-

[1] Eine umfangreiche Auflistung verschiedener Wiki-Systeme findet sich unter `http://c2.com/cgi/wiki?WikiEngines`.

[2] `http://www.pmwiki.org`

stellungen sowie Programmerweiterungen in Form so genannter *Recipes* beisteuern (vgl. [Lan07], S. 395).

Unsere Hochschule hat sich nicht zuletzt auch deswegen für PmWiki als zentral installierte Wiki-Variante entschieden, weil es anders als etwa beim von der Wikipedia eingesetzten MediaWiki Techniken zur Dokumentstrukturierung gibt. Während Seiteninhalte beim MediaWiki lediglich relativ lose über Stichwörter referenziert werden können, lassen sich in PmWiki mit Hilfe speziell aufgebauter Link-Listen, den *Wiki-Trails*, beliebige Gliederungstrukturen, z.B. Inhaltsverzeichnisse mit Kapiteln und Unterkapiteln, realisieren. Dies war eine wesentliche Voraussetzung für die geplanten Einsatzszenarien von media2mult.

Seine Flexibilität erreicht PmWiki aus Entwicklersicht im wesentlichen durch zwei Ansätze: zum einen basiert das System auf frei definierbaren Markup-Regeln und zum anderen lassen sich die Seiteninhalte auf Grundlage so genannter *Actions* beliebig auswerten.

2.1 Actions

Jedes Wiki-System muss als Mindestanforderung prinzipiell zwei verschiedene Sichten auf den Inhalt einer Wikiseite bieten: eine Bearbeitungssicht und eine Darstellungssicht. Während erstere den Seitencode in einem Textfeld zur Bearbeitung anzeigt, wertet die Darstellungssicht den Seitencode aus und schickt daraus generiertes HTML zum Browser. PmWiki realisiert die unterschiedlichen Sichten auf denselben Inhalt mit Hilfe frei definierbarer *Actions*, die in Form eines GET-Parameters übergeben werden. Jedem Aktionsbezeichner ist ein entsprechender Action-Handler zugeordnet. Dieser wird nach Aufruf einer Seite automatisch ausgeführt und realisiert so die gewünschte Aktion. Im Falle der Bearbeitungs- und Darstellungssicht sehen die Seitenaufrufe folgendermaßen aus, wobei die explizite Angabe der Aktion *browse* optional ist:

```
http://www.mein-wiki-server.de?n=Gruppe.Seite&action=edit
http://www.mein-wiki-server.de?n=Gruppe.Seite&action=browse
```

In beiden Fällen wird der Inhalt der Seite `Gruppe.Seite` ausgewertet und entsprechend aufbereitet an den Browser gesandt.

Weitere Beispiele für vordefinierte PmWiki-Aktionen sind *upload* zum Hochladen und Anhängen von Dateien sowie *source* zum direkten Abrufen des Seitencodes. Die Cross-Media-Publishing-Komponente von media2mult definiert eine weitere Aktion, welche die erforderlichen Seiteninhalte zusammenträgt und den Konvertierungsvorgang startet.

2.2 Markup-Regeln

Prinzipiell ist PmWiki nichts anderes als ein webbasierter Textersetzer, dessen Hauptaufgabe darin besteht, die Bestandteile der Eingabesprache in HTML-Bestandteile zu konvertieren. Die zentrale, von der Aktion *browse* aufgerufene PmWiki-Funktion heißt `MarkupToHTML`. Auf den ersten Blick ist sie dafür zuständig, den vom Anwender eingegebenen Code in HTML zu konvertieren. Doch dabei handelt es sich nur um das Standardverhalten. Letztlich wird die Konvertierung aber durch Iteration über ein Array mit

zahlreichen Ersetzungsmustern realisiert, d.h. die Umwandlung erfolgt durch wiederholtes, sequenzielles Anwenden regulärer Ausdrücke, ohne dass auf ein versehentliche Ersetzen bereits zuvor ersetzter Seitenbestandteile Rücksicht genommen wird. Letzteres muss ggf. explizit von den Ersetzungsfunktionen durch Maskierung der Fragmente unterbunden werden.

Das Markup-Array selbst wird durch so genannte *Markup-Regeln* aufgebaut, die jedes Mal ausgewertet werden, sobald der Browse-Handler aufgerufen wird. Eine Markup-Regel besteht aus einem Namen, einer relativen Position im Markup-Array, einem Suchpattern sowie einem Ersetzungsausdruck. Die Standard-Regel zum Erzeugen kursiver Texte z.B. sieht wie folgt aus:

```
Markup("''", 'inline', "/''(.*?)''/",'<em>\$1</em>')
```

Der erste Parameter spezifiziert den Regelnamen – in diesem Fall zwei Apostrophe. Der zweite Parameter legt fest, an welcher Stelle die Regel in das Markup-Array aufgenommen werden soll. Bei dem Abschnitt *inline* handelt es sich um einen speziellen Ort, an dem alle Regeln gesammelt werden, die nur einzeilige Ersetzungen bewirken. Unter Angabe des Regelnamens können aber auch zuvor definierte Regeln referenziert werden. Die Positionsangabe `<''` etwa könnte dazu verwendet werden, um eine weitere Regel vor der oben angegebenen einzuordnen, so dass diese beim Konvertierungsvorgang früher zur Anwendung kommt.

Der dritte Parameter spezifiziert das Suchmuster und der vierte den Ersatztext. Zusammenfassend sorgt dieses Markup also dafür, dass Textbausteine der Form `''Text''` zu geeigneter Zeit in das HTML-Fragment `<em>Text</em>` überführt werden.

Erweiterungen an der PmWiki-Syntax können folglich relativ einfach durch das Hinzufügen weiterer Markup-Regeln realisiert werden. Ferner ist es möglich, den vordefinierten Regelsatz durch einen komplett anderen zu ersetzen, um beispielsweise XML- oder LaTeX-Code zu erzeugen. In diesen Fällen würde ein Aufruf der oben erwähnten Funktion `MarkupToHTML` also kein HTML erzeugen, so dass ihr Name die vielfältigen Möglichkeiten ein wenig verschleiert.

Die erste media2mult-Version wurde auf genau diese Weise realisiert. Für die PDF-Konvertierung wurde ein separater Regelsatz geschrieben, der aus dem Eingabecode DocBook-XML[3] generierte und diesen anschließend an die DocBook-Stylesheets[4] von Daniel Veillard weiterreichte. Es stellte sich jedoch heraus, dass das Doppeln sämtlicher Regeln das Einspielen aktuellerer PmWiki-Versionen aufgrund variierender Regelsätze erschwerte. Darüber hinaus erwies sich die bei DocBook angestrebte strikte Trennung zwischen Inhalt und Layout (vgl. [WM99], S. 36) im Wiki-Kontext als nicht ausreichend tragfähig, denn die Anwender waren von der Unterdrückung eines Großteils ihrer auf das Layout bezogenen Textauszeichnungen in den PDF-Dateien enttäuscht. Aus diesem Grund wird seit der zweiten Version ein anderes, weiter unten beschriebenes Konvertierungskonzept umgesetzt.

[3] `http://docbook.sourceforge.net`
[4] `http://wiki.docbook.org/topic/DocBookXslStylesheets`

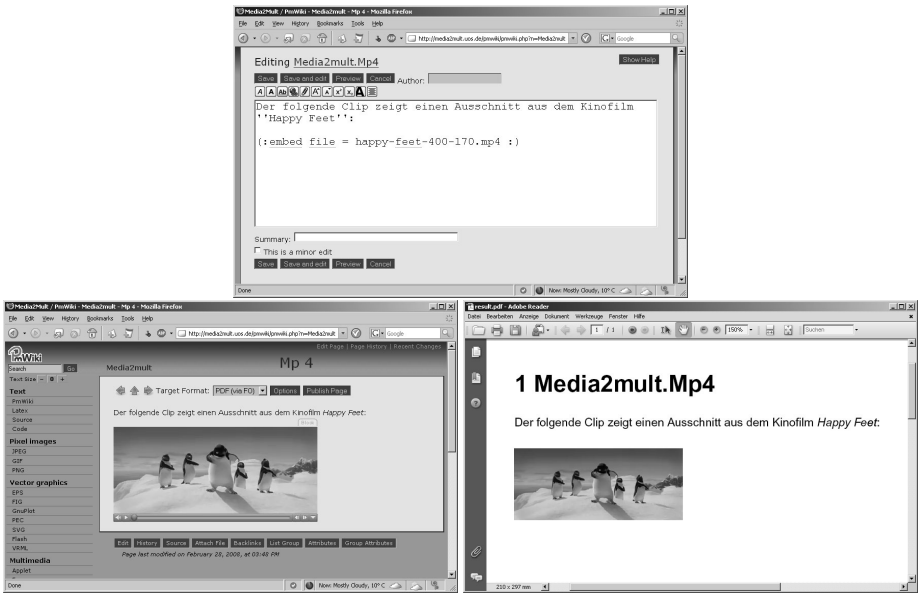

Abbildung 2: Aus dem eingebetteten MP4-Video wurde für die PDF-Fassung (unten rechts) ein Vorschaubild extrahiert.

3 Das erweiterte Markup von media2mult

Eine wesentliche Voraussetzung für die Akzeptanz eines neuen Autorenwerkzeugs ist die Unterstützung der von den traditionell eingesetzten Anwendungen bereit gestellten Funktionen, wie das Hinzufügen von Fußnoten sowie das Einbinden von Formeln und Grafiken. All diese Aspekte berücksichtigt ein Wiki-System in aller Regel nicht – insbesondere wenn im Hinblick auf die PDF-Generierung vektorielle Varianten der Dokumentbestandteile benötigt werden.

Das Plug-In media2mult stellt dem Anwender deshalb das neue Markup-Element *embed* zur Verfügung, mit dessen Hilfe eine Vielzahl unterschiedlicher Mediendateien in die Wikiseiten integriert werden können. Zu den unterstützten Formaten gehören unter anderem EPS, FIG, SVG, SWF, VRML, WAV, MP3, MIDI, MOV, MP4, FLV und einige mehr. Die einzubettende Mediendatei muss lediglich mit PmWikis Upload-Funktion auf den Server transferiert werden und kann anschließend z.B. via

```
(:embed file=grafik.eps:)
```

referenziert werden. Die Konvertierungsroutinen von media2mult sorgen automatisch dafür, dass ein für das jeweilige Ausgabemedium erforderliche Format erzeugt wird. Im Falle einer EPS-Datei wird für die Web-Darstellung beispielsweise ein PNG-Pendant generiert, beim PDF-Export hingegen ein einbettbarer PDF-Ausschnitt.

Wie der vorangehenden Aufzählung zu entnehmen ist, können mit *embed* nicht nur Grafiken und Bilder sondern auch Video- und Audiodateien referenziert werden – Formate also,

die sich im Browser per Plug-Ins anzeigen lassen, für der Druckausgabe aber kein direktes Äquivalent besitzen. media2mult versucht in diesen Fällen automatisch Vorschaubilder zu extrahieren (vgl. Abb. 2). Bei Formaten, bei denen dies nicht gelingen kann wird ein Warnhinweis ausgegeben. Der Autor hat dann die Möglichkeit, eine Alternativdatei anzugeben, die bei der PDF-Erstellung verwendet wird:

```
(:embed file=audio.wav print-file=bild.gif:)
```

Für die Einbindung von mathematischen Formeln stehen die bekannte LaTeX-Syntax zur Verfügung. Wie in LaTeX selbst, kann zwischen abgesetzten und in der Textzeile mitlaufenden Formeln unterschieden werden. Aus jeder Formel wird für die Web-Darstellung ein PNG-Bild produziert. Da dieser Konvertierungsvorgang den Seitenaufbau, insbesondere bei vielen eingebetteten Formeln deutlich verzögern kann, werden die Bilder nur einmal erzeugt, d.h. bei allen weiteren Seitenaufrufen entfällt die Wartezeit. Dazu wird ein einfaches MD5-basiertes Prüfverfahren verwendet. Dieses stellt sicher, dass bei Änderungen an den Formeln aktualisierte Grafiken produziert werden.

Die im Zusammenhang mit LaTeX zum Einsatz kommende Technik, Code-Schnipsel einer externen Eingabesprache in Grafiken umzuwandeln, wurde in media2mult verallgemeinert und kann so auf beliebige derartige Scriptprozessoren angewandt werden. Dies wurde exemplarisch für gnuplot, Metapost, POV-Ray und den Notengenerator p2i implementiert. Ähnlich wie bei den LaTeX-Formeln kann der Code entweder direkt als Bestandteil der Wikiseite eingegeben werden oder in einer separaten Datei hochgeladen werden. Letztere lässt sich wieder mit der erwähnten *embed*-Anweisung einbinden, wobei das Dateiformat anhand der verwendeten Dateiendung abgeleitet wird.

Neben der Unterstützung für Mediendateien bietet media2mult das Markup-Element *code* zur Darstellung von Quelltexten, die auf Wunsch automatisch zeilenweise durchnummeriert und mit farbiger Syntax-Hervorhebung versehen werden. Die Angabe der verwendeten Sprache für den Kolorierung erfolgt hierbei über ein optionales Attribut:

```
(:code file=bubblesort.java lang=java:)
```

Die Syntax zur Erzeugung von Fußnoten ähnelt der von LaTeX. Der gewünschte Text muss lediglich an der Stelle eingegeben werden, an der die Fußnotenziffer erscheinen soll. Der Text selbst wird am unteren Rand der Seite ausgegeben:

```
(:fn Dies ist eine Fußnote.:)
```

Da die Kapitel- und damit auch die Seitenfolge bei umfangreicheren Dokumenten durch Wiki-Trails festgelegt wird und keine inhärente Folge separater Wikiseiten existiert, beginnt die Fußnotennummerierung bei der Web-Darstellung im Gegensatz zum PDF-Export immer bei 1. In diesem Punkt weicht die Online-Version also vom druckbaren Dokument ab. Die Zuordnung zwischen Fußnotenziffer und -text bleibt jedoch stets erhalten, so dass keine Mehrdeutigkeiten entstehen können.

4 Technische Umsetzung der Medieneinbettung

Die automatische Verarbeitung der unterschiedlichen Mediendateien gehört zu den komplexeren Programmteilen von media2mult. Um die Erweiterbarkeit des Plug-Ins zu ge-

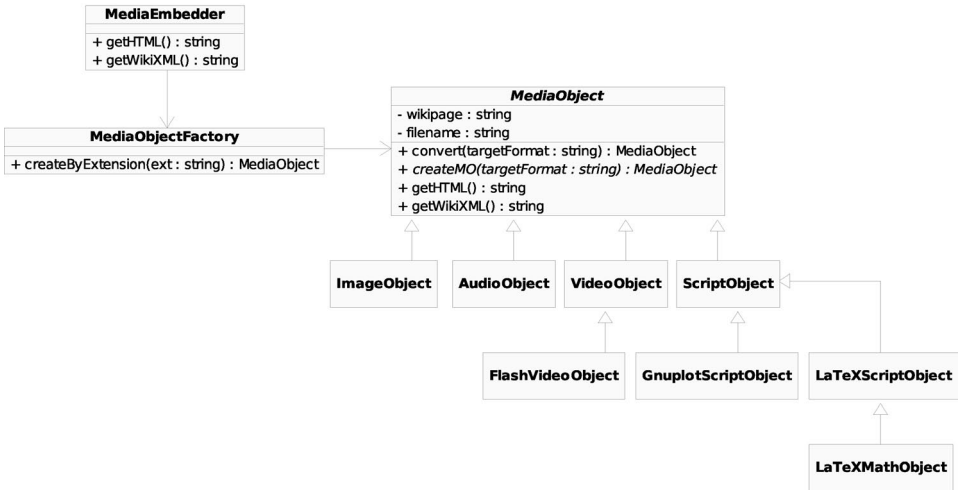

Abbildung 3: Ausschnitt aus der in media2mult verwendeten Klassenhierarchie zur einheitlichen Verarbeitung verschiedener Mediendateien.

währleisten und nicht für jedes neu zu unterstützende Format komplett neuen Code schreiben zu müssen, wurde eine vereinheitlichendes Klassenhierarchie entwickelt. Diese reduziert die erforderliche Code-Erweiterung bei später hinzugefügten Formaten auf ein Mindestmaß.

Wie in Abbildung 3 zu sehen, wurden die Dateiformate in die vier Kategorien *image*, *audio*, *video* und *script* unterteilt. Letztere umfasst alle Skriptsprachen, wie gnuplot oder LaTeX, die Bilder, Audios oder Videos beschreiben. Die jeweils zu den vier Kategorien gehörenden Klassen stammen von der Basisklasse `MediaObject` ab, welche u.a. die grundlegende Konvertierungsfunktion `convert` bereitstellt. Mit Hilfe von Template-Methoden wird hier der generelle Konvertierungsvorgang implementiert, der im Erfolgsfall ein neues, entsprechend der übergebenden Parameter erzeugtes Medienobjekt generiert. Die spezialisierten Methoden `createMO` der Kindklassen präzisieren die auszuführenden Schritte für den jeweiligen Medientyp.

Um die Methoden nutzen zu können, muss zunächst aus der in der *embed*-Anweisung angegebenen Ausgangsdatei ein passendes Medienobjekt erzeugt werden. Diese Aufgabe erledigt die Klasse `MediaObjectFactory`. Sie stellt unter anderem eine statische Methode bereit, die anhand der Dateiendung das Ausgangsformat ableitet und ein zugehöriges Objekt produziert. Wird anschließend dessen `convert`-Methode z.B. in der folgenden Form aufgerufen, entsteht aus dem zuvor erzeugten Videoobjekt ein neues Bildobjekt im PNG-Format:

```
$video = MediaObjectFactory.createByExtension('flv');
$image = $video->convert('png');
```

Für die zahlreichen Transformationen zwischen den unterschiedlichen Formaten greift media2mult auf inzwischen knapp 30 externe kostenlose Kommandozeilenprogramme

zurück, die aus Anwendersicht völlig transparent zum Einsatz kommen. Zu den verwendeten Programmen gehören unter anderem der SVG-Konverter *batik*, das Videowerkzeug *ffmpeg* sowie die beiden Tools *convert* und *mogrify* aus dem *ImageMagick*-Paket. Die aus den Konvertierungsprozessen hervorgehenden Dateien werden getrennt für jede Wikiseite in einem separaten Verzeichnisbaum abgelegt und von dem aus dem *embed*-Statement abgeleiteten HTML-Code referenziert.

Beispiel: Einbettung einer EPS-Datei

Zur Veranschaulichung der Arbeitsweise werfen wir nun einen Blick auf die Einbettung einer EPS-Datei in die Wikiseite. Wie erwähnt, muss der Anwender die Datei lediglich via Upload-Funktion auf den Server kopieren und mit dem *embed*-Statement auf der Wikiseite referenzieren:

```
Die folgende Grafik zeigt die Rendering-Pipeline von OpenGL:
(:embed file=rendering-pipeline.eps transparent=white:)
```

Die erste Zeile besteht in diesem Beispiel ausschließlich aus Text, der in der Darstellungssicht unverändert ausgegeben wird. In der zweiten Zeile findet der Textersetzer hingegen eine *embed*-Anweisung. Deren zugehörige Markup-Regel sorgt dafür, dass der Konvertierungsprozess gestartet wird.

Im ersten Schritt erkennt media2mult nun, dass es sich bei der einzubindenden Grafik um ein Format handelt, das vom Browser nicht direkt unterstützt wird und die Datei deshalb in ein Bitmap umgewandelt werden muss. Dazu wird das Kommandozeilentool *convert* aufgerufen:

```
convert -transparent white graph.eps graph.png
```

Der optionale Transparenzparameter wird an *convert* durchgereicht und sorgt hier dafür, dass alle weißen Bildbereiche transparent dargestellt werden. Im zweiten Schritt findet die eigentliche Textersetzung statt: das *embed*-Statement wird durch das HTML-Fragment `<img src="graph.png"/>` ersetzt.

5 Das Cross-Media-Publishing-Modul

Das Cross-Media-Publishing-Modul von media2mult stellt eine einfach zu bedienende Funktion bereit, die es dem Anwender ermöglicht, einzelne Wikiseiten oder beliebige Seitenfolgen nach PDF, PostScript oder RTF zu konvertieren. Bevor eine solche Funktion aber implementiert werden kann, muss die Frage nach der Ordnung der Seiten beantwortet werden, denn im Gegensatz zu gedruckten Dokumenten handelt es sich bei Wikifeldern quasi um digitale Loseblattsammlungen, deren Bestandteile sich durch keine eindeutige inhärente Ordnungsstruktur sequenziell anordnen lassen.

5.1 Wiki-Trails

Möchte man die Inhalte eines Wikifeldes aber in ein Druckformat gießen, so muss man zwangsläufig eine lineare Ordnung der Kapitel und Unterabschnitte festlegen, welche gleichzeitig auch zur schrittweisen Navigation durch die Wikiseiten genutzt werden kann. Die Definition einer solchen Ordnung wird in PmWiki mit Hilfe so genannter *Trails* realisiert. Bei einem Wiki-Trail handelt es sich um eine Aufzählungsliste, deren Einträge ausschließlich aus Referenzen auf Wikiseiten bestehen. Durch das Einfügen von Unterpunkten kann eine mehrstufige Gliederungsstruktur ähnlich der eines Inhaltsverzeichnisses aufgebaut werden. In PmWiki-Syntax hat ein Trail etwa folgende Gestalt und führt zu der nebenstehenden Struktur:

```
* [[Einleitung]]
* [[Sortierverfahren]]
** [[Bubblesort]]
** [[Quicksort]]
* [[Suchalgorithmen]]
```

1. Einleitung
2. Sortierverfahren
   2.1 Bubblesort
   2.2 Quicksort
3. Suchalgorithmen

Findet media2mult auf einer Wikiseite einen Trail, dann erscheint in der Darstellungssicht der Seite ein *Publish Trail*-Button, mit dem der Konvertierungsprozess gestartet werden kann. Auf allen anderen Seiten fehlt er und es wird lediglich die Möglichkeit zur Konvertierung der aktuellen Einzelseite angeboten.

5.2 XML-basierte Konvertierung

Beim Klick auf den *Publish Trail*-Button wird die Aktion *m2m-publish* ausgelöst und sämtliche Inhalte der im Trail referenzierten Seiten in der festgelegten Reihenfolge eingesammelt. Im Anschluss daran wird der gesamte so zusammengetragene Seitencode mit Hilfe der im Abschnitt 2.2 erwähnten Funktion `MarkupToHTML` in eine WikiXML-Datei, die aus erweitertem XHTML besteht, übersetzt. Dazu werden für den Konvertierungsprozess spezielle Markup-Regeln geladen, die dafür sorgen, dass zusätzliche Informationen zur Dokumentstruktur und den eingebetteten Mediendateien generiert werden.

Die Entscheidung für XHTML als Grundlage für alle weiteren Konvertierungsschritte hat gegenüber dem in der Vorgängerversion favorisierten Weg über DocBook den entscheidenden Vorteil, dass sich sämtliche Markup-Regeln, die einerseits standardmäßig vom PmWiki-System definiert und andererseits über zusätzliche Plug-Ins hinzugefügt werden, unverändert nutzen lassen. Es sind insbesondere keine Dopplungen aller elementaren Regelsätze erforderlich, welche die Erweiterung und Aktualisierung des Systems deutlich erschweren würden. Lediglich die neuen, von media2mult eingeführten Anweisungen müssen, je nachdem, ob sie für die HTML-Darstellung der Wikiseiten oder dem WikiXML-Export ausgewertet werden, unterschiedlich gehandhabt werden.

Auch wenn die DocBook-Variante verworfen wurde, stellte sich der generelle Einsatz von XML-Technologien als außerordentlich vorteilhaft heraus. Insbesondere der komplexe, aber sehr mächtige XSL-FO-Standard[5] ermöglicht das vollautomatische Erzeugen

[5] http://www.w3.org/TR/xsl/#fo-section

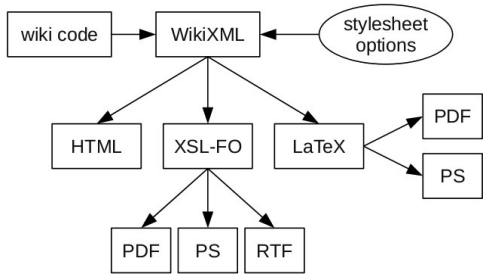

Abbildung 4: Konvertierungsschritte vom Wiki-Eingabecode zu den verschiedenen Dokument-Zielformaten.

qualitativ hochwertiger Dokumente. Ein wesentlicher Vorteil gegenüber LaTeX – welches üblicherweise die erste Wahl darstellt, wenn es im wissenschaftlichen Umfeld um professionellen Textsatz geht – ist die bessere Unterstützung der Netzwerkkommunikation. So lassen sich sehr leicht auf anderen Servern lagernde Dateien referenzieren, ohne diese zuvor explizit herunterladen zu müssen.

Darüber hinaus stellt u.a. das starre Tabellen-Handling von LaTeX ein relativ großes Problem dar. Es ist nahezu unmöglich, eine vorgegebene HTML-Tabelle ohne vorangehende Analyse des Tabelleninhalts in ein LaTeX-Pendant zu übersetzen. Andernfalls ragt die Tabelle entweder über den rechten Seitenrand hinaus, es entstehen trotz kurzer Texte zu breite Spalten oder trotz langer Textzeilen zu schmale Spalten. In jedem Fall ist eine individuelle Optimierung jeder Tabelle unumgänglich. Insgesamt hat sich das Compile-Edit-Revise-Paradigma von TeX im Bereich der automatischen *One-Pass*-Dokumentgenerierung als nur bedingt tragfähig erwiesen. XSL-FO wurde speziell für diesen Einsatzbereich entwickelt und arbeitet folglich zuverlässiger, obgleich auch hier noch einige Optimierungsspielräume bestehen.

Die XSL-FO-Dateien werden ausgehend von der vom Wiki-System erzeugten WikiXML-Datei mit Hilfe eines parametrisierten XSLT-Skripts erzeugt. Bei der Konzeption der Transformationsregeln wurde versucht, eine Vielzahl der Layout-Bestandteile zu übertragen, so dass die generierten Dokumente dem Layout der Wikiseiten möglichst nahe kommen. Das gilt nicht nur für einfache Textauszeichnungen sondern u.a. auch für seitlich vom Text umflossene Grafiken, Tabellenformatierungen sowie Absatzeinzüge. Zu große Bilder, die über mindestens einen Seitenrand hinausragen würden, werden vom Skript automatisch herunterskaliert, so dass sie anschließend auf die Seite passen.

5.3 Ausgabeformate

Die meisten der zur Zeit verfügbaren FO-Prozessoren unterstützen bereits mehrere Ausgabeformate, so dass die Hauptarbeit mit dem Erzeugen der FO-Datei abgeschlossen ist. Die Prozessoren erledigen anschließend die verbleibende Arbeit. Aus diesem Grund wurde in die Entwicklung der FO-Ausgabe die meiste Zeit investiert. Dennoch werden darüber hinaus noch zwei weitere Wege verfolgt (vgl. Abb. 4).

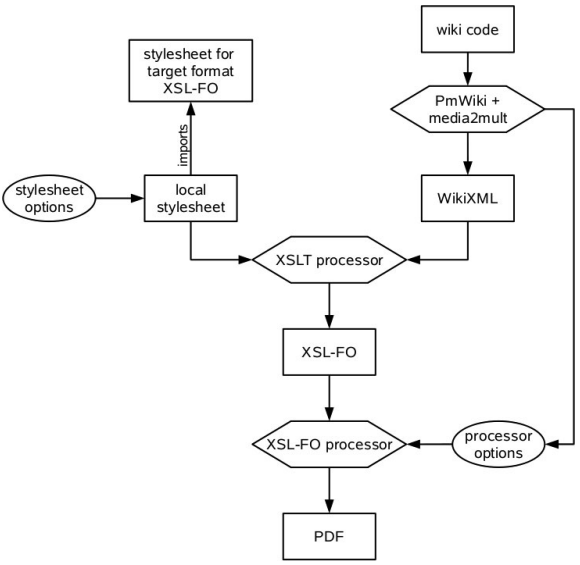

Abbildung 5: Schematische Darstellung der Cross-Media-Publishing-Komponente von media2mult.

Zum einen wird ein Stylesheet zur Erzeugung von statischen, untereinander verlinkten HTML-Seiten entwickelt, welche unabhängig von PmWiki eingesetzt und beispielsweise auf CD gebrannt werden können. Zum anderen möchten wir, trotz der oben genannten Nachteile, den Konvertierungsweg über LaTeX nicht ignorieren, denn für viele Anwendungsbereiche hat es sich bereits als vorteilhaft erwiesen, Content aus Wikifeldern automatisch nach LaTeX zu übersetzen, auch wenn der Code anschließend noch manuell optimiert werden muss. Über diesen Weg ist an unserer Hochschule bereits ein Buch entstanden, das von mehreren Autoren in PmWiki geschrieben, anschließend nach LaTeX konvertiert und schließlich vom Verlag veröffentlicht wurde.

5.4 Stylesheet-Optionen

Das von den XSLT-Stylesheets erzeugte Layout ist nur zum Teil statisch vorgegeben. Über zahlreiche Parameter lassen sich verschiedene globale Layout-Varianten konfigurieren. Dazu gehören neben den Seitenformat- und Randeinstellungen auch Optionen zur automatischen Erzeugung von Fußnoten für externe Weblinks oder das Erzeugen von Bookmarks für PDF-Dateien.

Der Anwender kommt mit diesen Parametern nur indirekt über einen Konfigurationsdialog in Kontakt. Seine Einstellungen werden in eine lokale XSLT-Datei mit Variablendefinitionen überführt, die gleichzeitig als Wrapper für das eigentliche Stylesheet dient. Die Kombination aus WikiXML-Datei und lokalem Stylesheet führt nach der Verarbeitung durch einen XSLT-Prozessor z.B. zur XSL-FO-Datei, die anschließend in das gewünschte Zielformat konvertiert wird (vgl. Abb. 5).

6 Zusammenfassung

Ein mit media2mult erweitertes PmWiki-System gibt Autoren ein server-basiertes Werkzeug an die Hand, mit dem auf relativ einfache Weise kollaborativ multimediale Dokumente erstellt werden können, die auf Wunsch sofort im Web zur Verfügung stehen und ferner in diverse Printformate konvertiert werden können. Dazu stellt das System transparente Mechanismen zur Umwandlung verschiedenster Medienobjekte bereit. Diese ermöglichen u.a. das Einbinden von EPS-Dateien in Wikiseiten und die Extraktion von Vorschaubildern aus Videos.

Das Cross-Media-Publishing-Modul ermöglicht u.a. die Generierung von PDF- und RTF-Dateien. Dabei wird das auf XML beruhende Konvertierungsverfahren vollständig vor dem Anwender verborgen. Die Stylesheet-Parameter, mit denen sich globale Layout-Einstellungen vornehmen lassen, werden über ein Web-Formular gekapselt und ebenfalls transparent an den XSLT-Prozessor weitergereicht.

Insgesamt hat sich beim Einsatz des Systems an unserer Hochschule sowie der Fachhochschule gezeigt, dass Dozenten und Studierende, insbesondere auch der nicht-technischen Fachbereiche, nach sehr kurzer Einarbeitungszeit media2mult bedienen können und das Tool zunehmend in ihren Lehrveranstaltungen einsetzen.

Ein media2mult-Dokument mit den wichtigsten Syntaxbeispielen liegt zur Demonstration unter `http://www.media2mult.de` bereit.

Literatur

[EGH05] Anja Ebersbach, Markus Glaser und Richard Heigl. *WikiTools. Kooperation im Web*. Springer, 2005.

[HTV99] Viktor Herzog, Frank M. Thiesing und Oliver Vornberger. Der Dokumentgenerator mas2tex. In *Workshop neue Medien in Forschung und Lehre. GI-Jahrestagung*. Paderborn, 1999.

[HTVW] Viktor Herzog, Frank M. Thiesing, Oliver Vornberger und Carsten Wilhelm. *mas2tex – ein Generator für Postscript und HTML*. `http://www-lehre.inf.uos.de/mas2tex/artikel`.

[Lan07] Christoph Lange. *Wikis und Blogs*. Böblingen, 2007.

[LC01] Bo Leuf und Ward Cunningham. *The Wiki Way. Quick Collaboration on the Web*. Addison Wesley, Amsterdam, 2001.

[Mic05] Patrick Michaud. WikiWikiWebs and PmWiki. `http://www.pmichaud.com/2005/pres/michaud-dalphp-pmwiki-nov-8.pdf`, 2005.

[TG05] Tobias Thelen und Clemens Gruber. Textproduktions- und Kommunikationsprozesse in WikiWiki-Webs. In Hans-Werner Huneke, Hrsg., *Geschriebene Sprache: Strukturen, Erwerb, Modellbildung*, Seiten 183–202. Mattes Verlag, Heidelberg, 2005.

[WM99] Norman Walsh und Leonard Muellner. *DocBook. The Definitive Guide*. O'Reilly, 1999.

Orchestrierung von Web 2.0-Anwendungen im Kontext hochschulischer Lehr-/Lernprozesse

Angela Carell, Isabel Schaller

Informations- und Technikmanagement
Ruhr Universität Bochum
Universitätsstraße 150

angela.carell@rub.de
isabel.schaller@rub.de

Abstract: Web 2.0–Anwendungen werden zunehmend unter der Perspektive des „Technology Enhanced Learning (TEL)“ im Kontext computerunterstützten kooperativen Lernens eingesetzt. Der Beitrag stellt die wesentlichen Prinzipien des Web 2.0 vor und entwickelt auf der Grundlage der zentralen Aspekte kollaborativen Lernens Thesen zur Bedeutung des Web 2.0 für die technische Unterstützung verschiedener kollaborative Lehr-/Lernszenarien im Rahmen des TEL-Ansatzes. Anhand eines Beispielszenarios wird aufgezeigt, wie und in welcher Form Web 2.0 Anwendungen orchestriert werden können, um ein computerunterstütztes kooperatives Lehr-/Lernszenario in der Präsenzlehre zu unterstützen und welche Auswirkungen deren Einsatz auf den kooperativen Lernprozess der Studierenden haben wird.

1 Einleitung

Beim computerunterstützten Lernen wird heute wieder – anders als in den ersten Jahren des E-Learningbooms - das Lernen selbst in den Mittelpunkt gerückt. Während früher die Entwicklung technischer Werkzeuge im Fokus stand und sich die Entwicklung von Lehr-/Lernszenarien an den Möglichkeiten (und Grenzen) der Technik orientierten, steht nun die Frage im Zentrum, welche technischen Hilfsmittel in einem bestimmten didaktischen Szenario die intendierten Lernprozesse optimal unterstützen können. Im europäischen Forschungsraum hat sich dieser Perspektivwechsel sprachlich in der Formulierung des „Technology Enhanced Learning“[1] (TEL) niederschlagen. Er bietet dem Lehrenden die Möglichkeit, technische Anwendungen flexibel nach den jeweiligen didaktischen Anforderungen zusammenzustellen und in unterschiedlichen Phasen des Lehr-/Lernprozessen einzusetzen. Klassische Lernumgebungen können hier ebenso genutzt

[1] http://cordis.europa.eu/fp7/ict/telearn-digicult/home_en.html