Dokumentation der Werkzeuge für digitales Publizieren

Inhaltsverzeichnis

Kapitel 1

Projektziele, Design-Kriterien für eigene Software

Der wichtigste Grundsatz lautet **freie Lizenzierung**. Seit der allgemeinen Verfügbarkeit von Digitaltechnologie und der darin inherenten Entkoppelung der Information von ihrem physischen Träger bei gleichzeitig immenser Skalierbarkeit durch den Computer als Universalrechenmaschine (Interoperabilität) steht fest, dass das prä-digitale Urheberrecht in seiner gegenwärtigen Form dringend reformbedürftig ist, weil es mittlerweile infolge technischer, rechtlicher, sozialer und ethischer Probleme das genaue Gegenteil von dem erreicht, wofür es eigentlich gedacht war. Seit den 80er-Jahren tobt der Kampf um digitale Grundrechte in den Bereichen Software, Musik, Video und jetzt auch bei den Büchern, wo aber die Öffentlichkeit wenig organisierte Interessensvertretung hervorgebracht hat und der Gesetzgeber tendenziell eher den Vorschlägen restriktiver Rechteverwerter folgt.

Bei Software und manchen Formaten ist eine Form von DRM schon allein aus technischen Gründen gegeben, sodass die unabhängige, eigenständige Datenverarbeitung effektiv unmöglich gemacht werden kann. Die Anwendung des Urheberrechts auf Software trotz ihres Charakters als Werke von praktischem Nutzen tut ihr Übriges, um Nutzer und andere Entwickler in Abhängigkeit zu bringen. Die einzigste Chance besteht im Moment darin, dass die Urheber ihre Software frei lizenzieren, indem Nutzern umfangreiche Nutzungsrechte eingeräumt werden und anderen Entwicklern die kollaborative Zusammenarbeit ermöglicht wird. Darum ist unsere Software unter der GNU Affero General Public License 3[1] or any later version lizenziert, um die Verfügbarkeit, Veränderbarkeit, Weiterverbreitbarkeit und unabhängige Verwendbarkeit langfristig sicherzustellen.

Wenn es um Medienwerke wie Bücher geht, ist freie Lizenzierung genauso wie bei der zugrundeliegenden verarbeitenden Software eine elementare Voraussetzung für einen ganzheitlich digitalverträglichen Literaturbetrieb. Vielfältig muss das digitale Potential im Hinblick auf den Umgang mit Texten unausgeschöpft werden, weil immer noch das Urheberrecht dazu missbraucht wird, das Geschäftsmodell des Verkaufs einzelner Dateikopien als Nachbildung der Knappheit aus der physischen Welt einzuführen, obwohl der digitalen Welt der Überfluss zugrundeliegt und die künstliche Verknappung dem Anliegen und

[1] `https://www.gnu.org/licenses/agpl-3.0.html`

Zweck einer Veröffentlichung direkt entgegensteht („Veröffentlichung“ hat etwas mit Öffentlichkeit zu tun). Ziel muss es deshalb sein, die Arbeit an und mit frei lizenzierten sowie gemeinfreien Texten aktiv voranzutreiben, die dafür vorhandenen Geschäftsmodelle weiter zu etablieren und so etwas wie eine freie digitale Bibliothek in welcher Form auch immer entstehen zu lassen, die in den Genuss umfangreicher als auch vielfältiger Community- und Softwareunterstützung kommen wird.

Mehr zum Thema von Richard Stallman (über Software[2], über andere Medienwerke[3]), Cory Doctorow (über die Implikationen von Digitaltechnologie für das Konzept „Buch“[4], über DRM[5]) und The League Of Noble Peers (Steal This Film 2[6]).

Die nächste wesentliche Eigenschaft der Software in der Werkzeugkiste betrifft die **Automatisierung**. Es gibt einen ständig wachsenden Bestand an Texten und jeder dieser Texte bedarf umfangreicher Pflege, Anreicherung und Aufbereitung. Anstatt viel Zeit mit manueller Bearbeitung zu verschwenden, die bei der nächsten Gelegenheit schon wieder hinfällig werden kann, sollen wiederkehrende Arbeitsschritte möglichst automatisiert werden, damit mehr Zeit für automatisierungskompatible Anreicherungen bleibt. Die Automatisierung geht keineswegs zulasten der Qualität oder der Gestaltungsfreiheit, sondern kann im Rahmen der Verarbeitung durchaus Konfigurationseinstellungen oder die Einbindung externer Module zwecks Sonderbehandlung vorsehen, sodass der Workflow ein durchgängig datengetriebener ist und die Automatisierungssoftware lediglich die Methodik zur Verfügung stellt, um von einer definierten Eingabe zur gewünschten Ausgabe zu gelangen, womit der Prozess beliebig oft auf dieselben oder kompatible Eingabedaten neu angewendet werden kann.

Die technische Grundlage dafür stellt XML als universaler Standard zur Auszeichnung von textorientierten Daten dar, womit zahlreiche auf bestimmte Anwendungsbereiche spezialisierte Formate wie z.B. HTML, DocBook, ODT oder TEI definiert wurden, wofür in quasi allen Programmiersprachen Zugriffs- und Verarbeitungsbibliotheken zur Verfügung stehen, worauf mächtige Werkzeuge wie XML-Schema-Validierer, XSLT- und XSL-FO-Prozessoren operieren. XML sichert dabei die Interoperabilität, indem zwischen unterschiedlichen Formaten hin- und herkonvertiert werden kann, Nicht-XML-Quelldateien über Parser nach XML und XML-Daten über entsprechende darauf zugeschnittene Generatoren (evtl. irreversibel) in die gewünschten Zielformate überführt werden. Freilich sind Konzepte wie Single Source Publishing[7] oder gar „Multi Source Publishing“ (Erzeugen mehrerer Zielformate unter der Zusammenfügung mehrerer Datenquellen, die womöglich ihrerseits jeweils mehrfach konvertiert werden müssen) ganz im Sinne der Automatisierung. Allerdings soll die technische Komplexität der Formate, der Verarbeitung und der Abstimmung der Werkzeuge vor dem Benutzer durch grafische Oberflächen komplett verborgen werden (siehe Robert Cailliau zu den Grundlagen des World Wide Webs[8]), weil einerseits manuelle Eingriffe beim nächsten Durchlauf hinfällig wären und

[2] `https://archive.org/details/Richard.Stallman.Manchester.2008`
[3] `https://www.youtube.com/watch?v=eginMQBWII4`
[4] `https://www.youtube.com/watch?v=nZFg-uq5zBA`
[5] `https://www.youtube.com/watch?v=HUEvRyemKSg`
[6] `https://archive.org/details/StealThisFilmII`
[7] `https://en.wikipedia.org/wiki/Single_source_publishing`
[8] `https://www.youtube.com/watch?v=x2GylLq59rI`

andererseits die Erstellung qualitativer Quelldokumente und die Definition von Workflows mithilfe von sinnvollen Standardvorgaben stark vereinfacht werden können.

Die Automatisierung profitiert ganz erheblich von der **Modularität** der Software in der Werkzeugkiste. Die meiste Software für Self-Publisher folgt einer monolithischen Architektur, d.h. ein einziges großes Programm vereint sämtliche angebotene Funktionalität in sich selbst mit dem Ziel, eine integrierte Arbeitsumgebung bereitzustellen. Der Nachteil davon ist, dass einzelne Funktionen oft nicht separat aufgerufen werden können, sondern stattdessen immer die komplette Arbeitsumgebung gestartet werden muss. Dann ist die Arbeitsumgebung in der Regel eine grafische, sodass die Bedienung des Programms primär mit der Maus erfolgt. Bei der Automatisierung macht es aber nur wenig Sinn, Klicks auf Buttons und Menüs auszulösen, denn die Eingabe von Daten soll ja schließlich von außen parametrisiert angesteuert werden, um einen rein datengetriebenen Ablauf zu erreichen.

Dementsprechend soll die Software, die im Rahmen des Projekts entwickelt wird, aus einer Reihe von kleinen Einzelprogrammen bestehen, die zu größeren automatisierten Workflows zusammengeschaltet werden können. Einerseits entsteht dadurch eine hohe Flexibilität, weil die Programme auch in anderen Kombinationen eingesetzt werden können, andererseits können zwischen den einzelnen Verarbeitungsschritten externe Komponenten aufgerufen werden. Dass die Programme auch abseits automatischer Workflows zur Bewältigung begrenzter Aufgaben in einem ansonsten manuellen Aufbereitungsworkflow genutzt werden können, versteht sich von selbst. Dieser Ansatz schließt übrigens keineswegs aus, dass vor oder während der Verarbeitung umfangreiche benutzerspezifische Einstellungen hinterlegt werden können, die dann von den jeweiligen Einzelschritten berücksichtigt werden. Eine optionale grafische Oberfläche ist dabei behilflich, die überdies auch zur Steuerung und Bedienung der Einzelprogramme, der Workflows und des Gesamtsystems dient. Um die Einzelprogramme möglichst unabhängig von den sie aufrufenden Workflows zu halten, erfolgt die Kommunikation über wohldefinierte Schnittstellen, über welche die Workflows die in „Jobs“ zusammengefassten Einstellungen einspeisen können. Ein Nachteil dieses Ansatzes ist, dass die Komplexität der Abhängigkeiten bei Änderungen an den Schnittstellen oder in den Einzelprogrammen umso mehr steigt, je mehr Workflows sich der betroffenen Programme bedienen – dem soll nach Möglichkeit mit zusätzlichen Abstraktionsebenen begegnet werden, welche Schnittstellendetails nach oben hin verbergen (kapseln).

Die entwickelte Software besteht sowohl aus **Online- als auch Offline**-Teilen. Die Ausführbarkeit der Programme auf dem lokalen Rechner ist wichtig, um die Hoheit über die eigene Datenverarbeitung (siehe das Grundrecht auf Gewährleistung der Vertraulichkeit und Integrität informationstechnischer Systeme[9], dessen Verletzung ja nicht fahrlässigerweise begünstigt zu werden braucht) aufrecht erhalten zu können. Ausgenommen davon sind logischerweise Funktionen, die nur in einem Vernetzungs-Kontext Sinn machen wie z.B. eine Benutzerverwaltung, mithilfe welcher die Aufteilung von Aufgaben für kollaboratives Arbeiten an einem Projekt organisiert werden kann. Allerdings

[9] `https://de.wikipedia.org/wiki/Grundrecht_auf_GewÃ¤hrleistung_der_Vertraulichkeit_und_IntegritÃ¤t_informationstechnischer_Systeme`

können externe Online-Komponenten wie das MediaWiki[10] oder WordPress[11] von Online- und/oder Offline-Varianten unserer Programme angesteuert werden, wobei die Unterstützung von dezentralen Peer-to-Peer-Protokollen zwecks Bereitstellung derselben Funktionalität zu bevorzugen wäre. Ob die Einzelprogramme und Workflows mit einem Web-GUI versehen und online ausführbar gemacht werden sollen, hängt vom jeweiligen Nutzungsszenario ab, denn die Verwendung der Online-Version statt der Offline-Version muss im Gegensatz zur Bereitstellung von Diensten für kollaboratives Arbeiten oder für integrierte Plattformen nicht gerade herausgefordert werden.

Ein weiteres wichtiges Anliegen ist die **Plattform-Integration**. Statt Werke, Werkzeuge, Dienste, Veröffentlichungskanäle und Geschäftsmodelle für sich isoliert zu betrachten, können diese Einzelbausteine miteinander kombiniert werden zu ganzheitlich digitalverträglichen Ökosystemen, in welchen die Produzenten höher entlohnt werden bei gleichzeitig für den Konsumenten günstigeren, quantitativ mehr und qualitativ besseren Ergebnissen. Freilich gibt es keine feste Rollenverteilung in solchen Ökosystemen, jedermann kann gleichberechtigt jede beliebige Funktion den eigenen Fähigkeiten und Fertigkeiten entsprechend ausüben, auch in Kooperation und Kollaboration. Interoperabilität und Vernetzung sind probate Mittel, um die verschiedenen Teilnehmer näher zusammenzubringen. Digitalverträgliche Plattformen fördern die Produktion, Distribution und Rezeption von Werken und erschweren sie nicht künstlich, um Interessen einzelner Parteien zu schützen auf Kosten der ebenso berechtigten Interessen anderer Parteien. Größte Aufmerksamkeit gebührt der Vermeidung von Abhängigkeiten, welche durch Einseitigkeit ein bestehendes Ökosystem ins Ungleichgewicht, in die Instabilität stürzen können. Daher sollten besagte Plattformen flexibel genug konzipiert sein, um einzelne Komponenten austauschen oder den Benutzer vor die Wahl stellen zu können, welche davon eingesetzt werden sollen.

Eine **Diskriminierung** findet nicht statt. Inhaltliche, personelle, qualitative (nutzungsrechtliche und technische Anforderungen ausgenommen), quantitative, gestalterische, strategische und kommerzielle Präferenzen werden nicht besonders berücksichtigt, sondern sind den handelnden Personen eigenverantwortlich anheim gestellt.

[10] `https://www.mediawiki.org/wiki/MediaWiki/de`
[11] `https://wordpress.org/`

Kapitel 2

Installationsanweisungen

Neben der Möglichkeit, die Programme der Werkzeugkiste aus dem Quellcode zu kompilieren, werden unter publishing-systems.org/downloads.php[1] vorbereitete Download-Packages angeboten. Da für die Ausführung eine Java-VM benötigt wird, sollte mittels des Befehls

java -version

ermittelt werden, ob eine oder welche Java-VM auf dem System bereits installiert ist. Unterstützt wird Java 1.6 und höher. Weil die Verwendbarkeit der Werkzeugkiste für die 100% freien Distributionen des GNU-Betriebssystems[2] stets gewährleistet werden soll, insbesondere aber gNewSense[3] als Referenzsystem gelten kann, wäre, sofern noch nicht vorhanden, die Java Runtime Environment des OpenJDKs[4] mittels

apt-get install openjdk-6-jre

zu installieren. Nachdem das Download-Package mit der höchsten Versionsnummer heruntergeladen und entpackt wurde, ist das Shell-Script

$/workflows/run_setup1.sh

auszuführen, woraufhin anschließend die einzelnen Tools und Workflows direkt oder über das dazugehörige Shell-Script aufgerufen werden können.

Sobald PDFs erzeugt werden sollen, ist auch ein installiertes LaTeX[5]-Satzsystem erforderlich, dessen pdflatex-Programm aufgerufen wird. Mit

apt-get install texlive-latex-base

apt-get install texlive-latex-memoir

apt-get install texlive-lang-german

können die benötigten Pakete eingerichtet werden.

Die Werkzeugsammlung kann grundsätzlich auch in anderen Betriebssystem-Umgebungen als GNU eingesetzt werden. Die dafür notwendigen, oben beschriebenen Software-Komponenten sind infolge ihrer freien Lizenzierung auf beispielsweise Microsoft Windows oder Apple Mac OS X portiert worden, jedoch wird dringend von der Verwendung proprietärer Betriebssysteme als Grundlage für die Ausführung von portierten und an sich frei lizenzierten Programmen abgeraten, weil sonst von einer Hoheit über die eigene Datenverarbeitung und der Achtung digitaler Grundrechte ebenfalls keine Rede sein kann.

[1] `http://www.publishing-systems.org/downloads.php`
[2] `https://www.gnu.org/distros/free-distros.html`
[3] `http://www.gnewsense.org/`
[4] `http://openjdk.java.net/`
[5] `https://de.wikipedia.org/wiki/LaTeX`

Kapitel 3

Beschreibung der Tools

bommanager1 unter $/bommanager/bommanager1/ kann Byte Order Mark[1] einer Datei hinzufügen oder entfernen. Da die Tools alle auf UTF-8 beschränkt sind, wird nur die UTF-8-BOM unterstützt. Per Kommandozeilen-Argument wird die Eingabe-Datei, die Option „add“ oder „remove“ und dann die Ausgabe-Datei mitgeteilt. Eingabe- und Ausgabe-Datei können dieselbe sein.

csv2xml1 unter $/csv2xml/csv2xml1/ ist ein Programm, welches eine CSV-Datei[2] nach XML konvertiert. Das Trennzeichen zwischen den Werten kann über einen Parameter mitgegeben werden, außerdem werden mehrzeilige Felder über die Einklammerung in Anführungszeichen unterstützt. Die Namen der Tags in der resultierenden XML-Datei werden der ersten Zeile entnommen, welche von manchen Benutzern für Spaltenüberschriften bei der Bearbeitung in z.B. LibreOffice verwendet wird.

csv2xml2 unter $/csv2xml/csv2xml2/ ist nahezu identisch mit csv2xml1 mit dem Unterschied, dass die Konvertierungs-Vorgaben über eine Konfigurationsdatei gesteuert werden. In selbiger wird das CSV-Trennzeichen festgelegt als auch, welche CSV-Spalte welchen XML-Tagnamen erhalten soll. Auf diese Weise kann auch eine Filterung der CSV-Daten erreicht werden, wenn nicht alle Spalten mit einer XML-Tagnamen-Zuordnung versehen werden.

epub2html1 unter $/epub2html/epub2html1/ leistet wenig mehr, als eine EPUB2-Datei zu entpacken. Obwohl eigentlich ein Zip-Programm für die Bewältigung dieser Aufgabe völlig ausreichen würde, hat epub2html1 den Effekt, dass die internen Dateien zur Organisation des EPUB-Containers nicht in den Ausgabe-Ordner übernommen werden und überdies eine flache Dateistruktur entsteht, wie sie für statische Webseiten typisch ist. Die zu entpackende EPUB-Datei sowie das Ausgabe-Verzeichnis werden in einer Konfigurationsdatei festgelegt. Im Ausgabe-Verzeichnis erstellt epub2html1 zwar keine index.html, jedoch wird eine index.xml generiert, welche die extrahierten XHTML-Dateien aufzählt, sodass aufrufende Software das Ergebnis weiterverarbeiten oder mittels XSLT eine index.html daraus abgeleitet werden kann. Das Programm unterstützt die automatische Ermittlung des DOCTYPEs noch nicht, sodass im Unterverzeichnis $/epub2html/epub2html1/entities/ die Datei config_xhtml1_1.xml nach config.xml kopiert werden muss, damit der Aufruf erfolgreich ist. Das Extraktions-Verzeichnis $/epub2html/epub2html1/temp/

[1] `https://de.wikipedia.org/wiki/Byte_Order_Mark`
[2] `https://de.wikipedia.org/wiki/CSV_(Dateiformat)`

→ epub_extraction/ wird nicht automatisch aufgeräumt.

epub2html1_config_create_new1 unter $/epub2html/epub2html1/ → workflows/ ist ein Programm zur Workflow-Unterstützung, welches eine Konfigurationsdatei für epub2html1 mit Standardvorgaben generiert, welche von aufrufender Software anschließend mit den tatsächlichen Pfaden überschrieben werden können.

file_picker1 unter $/gui/file_picker/file_picker1/ dient der Workflow-Unterstützung, indem ein Dateiauswahl-Dialog bereitgestellt wird, dessen anzuzeigende Dateitypen über eine Konfigurationsdatei festgelegt werden können. Neben der Dateitypen-Konfigurationsdatei kann das Verzeichnis, auf dem der Dialog zu Beginn stehen soll, per Kommandozeile mitgegeben werden. Ist eine Auswahl erfolgt, wird der absolute Pfad der ausgewählten Datei auf stdout ausgegeben. Abseits der Aufrufe durch andere Tools und Workflows gibt es keinen Verwendungszweck für file_picker1.

option_picker1 unter $/gui/option_picker/option_picker1/ dient der Workflow-Unterstützung, indem in Form eines Dialogs über eine Drop-Down-Auswahlbox aus mehreren Einträgen eine Option ausgewählt werden kann. In der Konfigurationsdatei, die per Kommandozeilenargument mitgegeben werden muss, können die Anzeigenamen, längeren Beschreibungstexte und internen IDs der einzelnen Optionen festgelegt werden; letztere wird bei Bestätigung des Dialogs auf stdout ausgegeben. Überdies muss die Beschriftung des Dialogfensters hinterlegt werden. Abseits der Aufrufe durch andere Tools und Workflows gibt es keinen Verwendungszweck für option_picker1.

html2epub1 unter $/html2epub/html2epub1/ packt XHTML-1.0-Strict-Dateien zu EPUB2 zusammen. Die internen Dateien zur Organisation des EPUB-Containers werden dabei automatisch generiert. Über eine Konfigurationsdatei können die Eingabe-XHTML-Dateien mit dem Titel, unter welchem sie in der EPUB-Navigation (Inhaltsverzeichnis) aufgeführt werden sollen, spezifiziert werden ebenso wie das Ausgabe-Verzeichnis, in welchem bei Erfolg eine out.epub abgestellt wird. Das Ausgabe-Verzeichnis muss bereits existieren, außerdem werden dort die Bestandteile des EPUBs ebenfalls abgelegt. Ferner können in der Konfigurationsdatei die Metadaten des EPUBs angegeben sowie eine Validierung der Eingabedateien deaktiviert oder wieder aktiviert werden, letzteres ist dringend zu empfehlen.

html2epub1_config_create_new unter $/html2epub/html2epub1/ → gui/html2epub1_config_create_new/ ist ein Programm zur Workflow-Unterstützung, welches eine Konfigurationsdatei für html2epub1 generiert. Zwingend erforderliche Angaben wie die Eingabe-XHTML-Dateien, das Ausgabe-Verzeichnis oder die Metadaten-Felder bleiben leer, sodass aufrufende Software die entsprechenden Werte vor Verwendung erst noch hinterlegen muss. Der Pfad für die Konfigurationsdatei kann, sofern der Dateiauswahldialog nicht aufgerufen werden soll, auch per Parameter mitgeteilt werden.

html2epub1_config_file_setup unter $/html2epub/html2epub1/gui/ → html2epub1_config_file_setup/ stellt einen Dialog bereit, mithilfe welchem eine per Parameter oder per Dateiauswahldialog angegebene html2epub1-Konfigurationsdatei hinsichtlich der beteiligten Datei- und Verzeichnispfade bearbeitet werden kann. So können XHTML-Dateien hinzugefügt, entfernt oder in ihrer Reihenfolge anders angeordnet werden. Deren Titel, welche für die EPUB-Navigation (Inhaltsverzeichnis) herangezogen werden, können editiert werden. Das Ausgabeverzeichnis kann hinterlegt und geändert werden. Ein

Button „Check“ dient der Überprüfung der Angaben, deren Korrektheit mittels kleiner Icons signalisiert wird. Wenn alle Icons grün sind, kann mit „Save“ gespeichert werden. Ein Verlassen mit „Exit“ speichert die Änderungen nicht!

html2epub1_config_metadata_editor unter $/html2epub/html2 → epub1/gui/html2epub1_config_metadata_editor/ stellt einen Dialog bereit, mithilfe welchem die Metadaten einer per Parameter oder per Dateiauswahldialog angegebene html2epub1-Konfigurationsdatei hinterlegt und bearbeitet werden können. Ein Button „Check“ dient der Überprüfung der Angaben, deren Korrektheit mittels kleiner Icons signalisiert wird. Wenn alle Icons grün sind, kann mit „Save“ gespeichert werden. Ein Verlassen mit „Exit“ speichert die Änderungen nicht!

html2epub1_config_merge1 unter $/html2epub/html2epub1/workflows/ führt zwei html2epub1-Konfigurationsdateien zusammen, wobei von der ersten per Parameter angegebenen Konfigurationsdatei die Dateihinterlegungen der XHTML-Eingabedateien sowie das Ausgabeverzeichnis und von der zweiten per Parameter angegebenen Konfigurationsdatei die Metadaten übernommen werden, um sie in die mit dem dritten Parameter angegebene Datei rauszuschreiben.

html2wordpress1 unter $/html2wordpress/html2wordpress1/ übermittelt eine XHTML-Datei an eine WordPress-Installation. Eine Internet-Verbindung ist erforderlich, ebenso muss seitens der WordPress-Installation die XML-RPC-Schnittstelle aktiviert sein, was aber standardmäßig der Fall ist seit WordPress 3.5. Um das Passwort des WordPress-Benutzers nicht im Klartext übermitteln zu müssen, muss serverseitig das „Secure XML-RPC“-Plugin[3] von Eric Mann (Version 1.0.0, das ist Commit 31) installiert sein. Über eine Steuerungsdatei wird die zu übermittelnde XHTML-Datei und der URL der XML-RPC-Schnittstelle angegeben. Der Public- und Private-Key für die Benutzer-Authentifizierung kann im WordPress-Benutzerprofil für html2wordpress1 generiert werden und muss ebenfalls in die Steuerungsdatei eingetragen werden. Damit die neu generierten Keys auch abgespeichert werden, den Klick auf „Update Profile“ nicht vergessen. Der Private-Key sollte der Geheimhaltung unterliegen, weshalb darauf zu achten ist, dass dieser nicht in der Steuerungsdatei verbleibt, wenn auch andere Computerbenutzer lesenden Zugriff auf die Datei haben. Eine Möglichkeit besteht darin, nach der Übermittlung das Paar aus Public- und Private-Key im Benutzerprofil zu verwerfen. Der Text der XHTML-Datei wird im Klartext übermittelt. Ferner muss bzw. kann in der Steuerungsdatei die ID des entsprechenden Benutzers, welchen Post-Typ die übermittelte Ressource aufweisen soll (üblich sind „post“ und „page“, aber auch Custom Post Types können angewendet werden), mit welchem Status die Ressource versehen werden soll (diverse Stati wie z.B. „publish“, „pending“, „draft“, „future“ oder „private“ sind üblich), Titel und Kurzzusammenfassung, Datum und Uhrzeit (hiermit lässt sich auch eine automatische zukünftige Veröffentlichung realisieren), das Darstellungsformat, der Slug (leserlicher Namen für den URL), die Aktivierung oder Deaktivierung von Kommentaren und Pingbacks, das Sticky-Flag (angepinnte Ressource), eine Thumbnailbild-ID, die ID einer Parent-Ressource, Custom Fields (werden normalerweise für Metadaten genutzt), IDs der Einträge einer hierarchischen Taxonomie (müssen vorhanden sein) und die Namen einer Tag-Taxonomie (müssen nicht vorhanden sein,

[3] `https://wordpress.org/plugins/secure-xml-rpc/`

werden angelegt) festgelegt werden. Es findet eine Validierung der Eingabe-XHTML-Datei statt. html2wordpress1 beschränkt sich auf den Inhalt von <body> der Eingabe-XHTML-Datei, welcher unverändert übermittelt wird, dabei aber auch CSS-Angaben in <head> samt den dort enthaltenen Metadaten verloren gehen. Die Antwort des Servers wird direkt auf stdout ausgegeben.

html2wordpress1_jobfile_create_new1 unter $/html2wordpress/ → html2wordpress1/html2wordpress1_jobfile_create_new1/ dient der Workflow-Unterstützung, welches eine Steuerungsdatei für html2wordpress1 mit Standardvorgaben generiert, welche von aufrufender Software anschließend mit den tatsächlichen Angaben überschrieben werden können. Der Pfad für die Steuerungsdatei kann, sofern der Dateiauswahldialog nicht aufgerufen werden soll, auch per Parameter mitgegeben werden.

html_attributeanalyzer1 unter $/html_attributeanalyzer/html_attri → buteanalyzer1/ liest aus einer XHTML-Eingabedatei alle enthaltenen Attribute, deren Wert, deren Element sowie das Eltern-Element aus und schreibt diese in eine Ausgabe-XML-Datei. Jede Kombination aus Attribut-Name, Attribut-Wert, Attribut-Element und Eltern-Element wird nur einmalig ausgewiesen. Das Programm wird bisher in keinem Workflow eingesetzt.

html_attributereplace1 unter $/html_attributereplace/html_attribute → replace1/ ersetzt Werte von Attributen in einer XHTML-Datei aufgrund einer Mapping-Konfigurationsdatei, wo die zu ersetzenden Attributwerte über den Element- und Attributnamen spezifiziert werden. Das Programm wird bisher in keinem Workflow eingesetzt.

html_concatenate1 unter $/html_concatenate/html_concatenate1/ fügt mehrere XHTML-Dateien aneinander und schreibt das Ergebnis in eine XHTML-Ausgabedatei. Die Steuerung eines Aufrufs erfolgt über eine Konfigurationsdatei, in welcher auch festgelegt werden kann, aus welcher XHTML-Datei die HTML-Kopfdaten übernommen werden sollen, wobei sich die Aneinanderfügung der referenzierten Eingabedateien immer auf den Inhalt des <body>-Tags bezieht. Die Datei, aus welcher die HTML-Kopfdaten übernommen werden sollen (inklusive <body>), muss nicht Teil der Liste der aneinanderzufügenden XHTML-Dateien sein. Wenn keine HTML-Kopfdatendatei angegeben wird, werden die Kopfdaten der ersten angegebenen XHTML-Datei herangezogen.

html_flat2hierarchical1 unter $/html_flat2hierarchical/html_flat2hier → archical1/ sorgt für sog. „Positional Grouping“. XML strukturiert Daten in aller Regel hierarchisch, wodurch ineinander verschachtelte Einheiten semantisch abgebildet werden. Es gibt jedoch auch die Möglichkeit, den Beginn (und theoretisch auch das Ende) logischer Einheiten über „Marker“ (manchmal auch „Milestones“ genannt) zu kennzeichnen, die auf derselben hierarchischen XML-Ebene stehen als ihre „Unterelemente“, für welche sie die umgebende Klammer darstellen sollen. Marker werden häufig verwendet, wenn sich überlappende Einheiten abgebildet werden sollen, was aber von html_flat2hierarchical1 nicht berücksichtigt wird. Ein anderer Verwendungsfall ist anzutreffen z.B. im ODT-Format, wo im Textverarbeitungsprogramm LibreOffice keine semantische Verschachtelung von Dokumentelementen hinterlegt werden kann, sodass intern nur eine „flache“ Struktur, sprich. eine Aufzählung der enthaltenen Elemente in der Reihenfolge ihres Auftretens, vorzufinden ist. Über eine Konfigurationsdatei kann html_flat2hierarchical1 angewiesen werden, solche Marker, die per Elementname und optional zusätzlich gemäß bestimmtem Attribut-Name und -Wert, versehen mit einer Angabe über deren hierarchische Wertigkeit, zu

Parent-Elementen der darauffolgenden Elemente umstrukturiert werden, was sämtliche Unterelemente einschließt, bis der nächste Marker gleicher oder höherer hierarchischer Wertigkeit vorgefunden wird, woraufhin sämtliche vorhergehenden, noch geöffneten Marker-Verschachtelungsebenen geschlossen werden. Das Programm unterstützt die automatische Auflösung der XHTML-DTDs noch nicht, sodass im Unterverzeichnis $/html_flat2hierarchical/html_flat2hier → archical1/entities/ die Datei config_xhtml1-strict.xml nach config.xml kopiert werden muss, damit der Aufruf für XHTML 1.0 Strict erfolgreich sein kann.

html_split1 unter $/html_split/html_split1/ teilt eine XHTML-Eingabedatei in mehrere Ausgabedateien auf, wobei die Aufteilungskriterien in einer Konfigurationsdatei spezifiziert werden. Als Kriterium wird ein Elementname und optional zusätzlich Attribut-Name und -Wert herangezogen. html_split1 berücksichtigt immer nur das zuerst gefundene Element, welches einem Kriterium entspricht, bis zu dessen Ende. Darin enthaltene Unterelemente, auf welche ebenfalls Kriterien zutreffen, werden nicht weiter unterteilt, wobei html_split1 zu diesem Zweck erneut und evtl. rekursiv auf die Ergebnis-Dateien aufgerufen werden könnte. Die Ergebnis-Dateien werden in einem Ordner als XML-Dateien abgelegt, da der ausgeschnittene Bereich nicht mit HTML-Kopfdaten versehen wird, welche aber per XSLT nachträglich wieder ergänzt werden können. Das Ausgabe-Verzeichnis wird mit einer Datei info.xml versehen, welche die erzeugten Aufteilungs-Dateien auflistet.

html_prepare4latex1 unter $/latex/html_prepare4latex1/ dient dem Escapen von LaTeX-Sonderzeichen in einer XHTML-1.0-Strict-Datei. Ausgenommen davon sind die Zeichen <, > und &, weil diese zwecks XML-Wohlgeformtheit in der XHTML-Datei weiterhin als Entity erhalten bleiben müssen, sodass sich nach einer Transformation von XHTML nach LaTeX ein anschließendes Suchen-und-Ersetzen anbietet, wo & durch \& ersetzt werden sollte. Bei < und > kann davon ausgegangen werden, dass die Transformation nach LaTeX als Nicht-XML-Ausgabeformat ein Deescaping der XML-Entities vornimmt. $/html2latex/html2latex1/layout/txtreplace1_replacement_dictionary_post → _preparation.xml stellt eine entsprechende Ersetzungs-Konfigurationsdatei für txtreplace1 bereit. Das Programm unterstützt die automatische Auflösung der XHTML-DTDs noch nicht, sodass im Unterverzeichnis $/latex/html_pre → pare4latex1/entities/ die Datei config_xhtml1-strict.xml nach config.xml kopiert werden muss, damit der Aufruf erfolgreich sein kann.

xml_prepare4latex1 unter $/latex/xml_prepare4latex1/ verhält sich wie html_prepare4latex1 mit dem Unterschied, dass keine spezifische Behandlung für XHTML-Dateien erfolgt (DOCTYPE wird weder berücksichtigt noch in die Ausgabe übernommen), sondern LaTeX-Sonderzeichen-Escaping für beliebige XML-Dateien vorgenommen wird. Die Konfigurationsdatei für die automatische Auflösung von DTD-Dateien (welche Definitionen von XML-Entities enthalten können) muss per Kommandozeilen-Argument mitgegeben werden.

odt2html1 unter $/odt2html/odt2html1/ konvertiert eine ODT-Datei (Textverarbeitungs-Dokumentenformat des OpenDocument-Standards, welcher u.a. OpenOffice/LibreOffice zugrundeliegt) nach XHTML 1.0 Strict. Neben der Eingabedatei muss ein Ausgabeverzeichnis per Parameter mitgegeben werden. Entweder wird im angegebenen Verzeichnis eine XHTML-Datei mit demselben Namen wie die Eingabedatei, aber mit der Dateiendung *.html angelegt, oder optional kann über einen dritten Parameter ein abweichender Dateiname gewählt werden. Im Ausgabe-

verzeichnis werden neben der XHTML-Datei auch die enthaltenen Bilder abgelegt, welche überdies dort in einer info.xml verzeichnet werden. Außer Bildern werden auch Links und Stil-Hinterlegungen zwecks semantischer Weiterverarbeitung in die Ausgabe übernommen. Das temporäre Extraktions-Verzeichnis $/odt2html/odt2html1/temp/ wird möglichst aufgeräumt, was jedoch nicht immer gelingt und dann mit einer nicht weiter relevanten Warnung gemeldet wird.

schemavalidator1 unter $/schemavalidator/schemavalidator1/ ist ein Wrapper, um mithilfe der Java-Standardlibrary eine XML-Schema-Validierung durchführen zu können, ohne ein externes Programm dafür aufrufen zu müssen. Neben der XML-Eingabedatei und dem Schema muss eine Konfigurationsdatei mitgeteilt werden, in welcher die DTDs der verwendeten XML-Entities hinterlegt sind sowie eine weitere Konfigurationsdatei, in welcher Imports in den Schemen aufgelöst werden. Sofern die XML-Eingabedatei oder das Schema keine externen Entities oder Schemen heranzieht, brauchen die Konfigurationsdateien auch keine Einträge aufweisen.

txtreplace1 unter $/txtreplace/txtreplace1/ nimmt eine einfache Textersetzung vor, wo die zu ersetzenden und ersetzenden Zeichen über eine Konfigurationsdatei festgelegt werden. Da die Konfigurationsdatei in XML verfasst ist, können XML-Tags in XML-Dateien damit nicht ersetzt werden, weil die Hinterlegung derselben mit dem Aufbau der Konfigurationsdatei konfligieren würde. Textinhalt von XML-Elementen kann aber in Eingabe-XML-Dateien wie in jeder anderen Dateiform auch ersetzt werden. Es wäre darauf zu achten, dass nicht versehentlich auch Tag-Namen mitersetzt werden, wo txtreplace1 natürlich keine Unterscheidung trifft zu gewöhnlichem Textinhalt innerhalb von XML-Tags. Die Einträge der Konfigurationsdatei werden der Reihe nach auf den Text angewendet, sodass spätere Einträge auf den bereits erfolgten Ersetzungen vorangegangener Einträge reagieren müssen.

unzip1 unter $/unzip/unzip1/ ist ein primitiver Wrapper, um mithilfe der Java-Standardlibrary ein Zip-Archiv entpacken zu können, ohne ein externes Programm dafür aufrufen zu müssen.

html2epub1-Workflow unter $/workflows/ ist für odt2epub1-Workflow fest auf eine Eingabe-XHTML-1.0-Strict-Datei abgestimmt, die auf $/odt2html/templates/template1/t basiert, um daraus ein EPUB2 zu erzeugen. Die Verwendung mit einer XHTML-Datei, die aus einem reinen odt2html-Aufruf stammt, ist nicht möglich, da odt2html1-Workflow gewisse Vorarbeiten vornimmt. Mittels html_split1 wird gemäß html_split1_config_part.xml und html_split1_config_chapter.xml aus $/odt2html/templates/template1/ in die einzelnen Bestandteile zerlegt, welche dann von xsltransformator1 entsprechend html2epub1_html_part.xsl bzw. html2epub1_html_chapter.xsl aus $/odt2html/templates/template1/ zum Ergebnis-Layout aufbereitet werden. Außerdem wird xsltransformator via $/odt2html/templates/template1/htm eine zusätzliche XHTML-Titelseite („Cover“) für das EPUB generieren. Mit $/odt2html/templates/template1/html2epub1_config.xsl wird xsltransformator1 eine html2epub1-Konfigurationsdatei anlegen, in welcher die Metadaten für das EPUB aus der html2epub1-Konfigurationsdatei, die per Parameter mitgeteilt wurde, per html2epub1_config_merge1 eingefügt werden. html2epub1 besorgt dann die Erstellung der EPUB2-Datei, die unter $/workflows/temp/epub/ als out.epub zu finden sein wird. Es wird versucht, den Inhalt des Verzeichnisses vorher rekursiv zu löschen. Die Dateien der Zwischenschritte unter $/workflows/temp/ werden nicht aufgeräumt.

html2pdf1-Workflow unter $/workflows/ erzeugt aus einer Eingabe-XHTML-

Datei, die auf $/odt2html/templates/template1/template1.ott basiert und durch odt2html1-Workflow vorbehandelt wurde, eine Ausgabe-PDF-Datei. Die XHTML-Datei kann entweder als Kommandozeilenargument mitgeteilt oder in einem Dialog ausgewählt werden. Der Workflow bereitet die Eingabedatei erst mit html_prepare4latex1 aus $/html2latex/html_prepare4latex1/ samt einem Aufruf von txtreplace1 aus $/txtreplace/txtreplace1/ mit $/html2latex/html2latex1/layout/txtreplace1_replacement_
→ _preparation.xml für die Aufbereitung durch LaTeX vor, um dann xsltransformator1 aus $/xsltransformator/xsltransformator1/ mit $/html2latex/html2latex1/layout/layout1.xsl die semantischen Stilkennzeichnungen gemäß der template1.ott an das LaTeX-Layout zu koppeln. Das Erebnis wird unter $/workflows/temp/pdf/ als out.tex abgelegt. Wenn pdflatex aufgerufen werden kann, findet sich dort dann auch eine entsprechende out.pdf. Es wird versucht, den Inhalt des Verzeichnisses vorher rekursiv zu löschen.

odt2all1-Workflow unter $/workflows/ erzeugt aus einer Eingabe-ODT-Datei, die auf $/odt2html/templates/template1/template1.ott basiert, HTML-, EPUB2- und PDF-Dateien, die in dem Verzeichnis abgelegt werden, welches per Kommandozeilenargument mitgeteilt wurde. Eine Steuerungsdatei, die ebenfalls per Kommandozeile referenziert wird, listet eine oder mehrere Eingabe-ODT-Dateien auf ebenso wie eine Konfigurationsdatei für html2epub1, aus welcher die Metadaten für das EPUB entnommen werden. Intern wird zuerst odt2epub3-Workflow und dann odt2pdf2-Workflow aufgerufen.

odt2all2-Workflow unter $/workflows/ ergänzt odt2all1-Workflow um grafische Dialoge zur Bedienung des Workflows, wo zuerst die odt2all1-Konfigurationsdatei ausgewählt oder neu angelegt wird, um dann mit odt2all1_config_edit2 aus $/workflows/gui/odt2all1_config_edit2/ die Eingabe-ODT-Dateien zu hinterlegen oder zu verändern. Daraufhin muss eine html2epub1-Konfigurationsdatei ausgewählt oder neu angelegt werden, welche anschließend mit html2epub1_config_metadata_editor aus $/html2epub/html2
→ epub1/gui/html2epub1_config_metadata_editor/ bearbeitet werden kann. Zuletzt muss noch ein Ausgabe-Verzeichnis ausgewählt werden, sodass odt2all1-Workflow unter Verwendung der hinterlegten Angaben aufgerufen werden kann.

odt2epub1-Workflow unter $/workflows/ ruft erst odt2html1-Workflow auf, erlaubt über html2epub1_config_metadata_editor aus $/html2epub/html2
→ epub1/gui/html2epub1_config_metadata_editor/ die Hinterlegung von Metadaten, um dann mit html2epub1-Workflow eine EPUB2-Datei unter $/workflows/temp/epub/out.epub zu erstellen. Es wird versucht, den Inhalt von $/workflows/temp/epub/ vorher rekursiv zu löschen.

odt2epub2-Workflow unter $/workflows/ unterscheidet sich von odt2epub1-Workflow nur dahingehend, dass statt der Eingabe der Metadaten des EPUBs per Dialog auf der Kommandozeile eine entsprechende html2epub1-Konfigurationsdatei und auch ein Pfad für das EPUB-Ergebnis mitgegeben werden kann.

odt2epub3-Workflow unter $/workflows/ nimmt aus einer Konfigurationsdatei mehrere hinterlegte Pfade zu ODT-Dateien und zu einer html2epub1-Konfigurationsdatei mit Metadaten entgegen, um daraus eine EPUB2-Datei zu generieren, die unter einem per Parameter angegebenen Pfad abgelegt wird. Hierzu wird odt2html1-Workflow auf jede Eingabedatei aufgerufen, welche dann mit html_concatenate1 zusammengefügt werden, html2epub1-Workflow packt dann zu EPUB2 zusammen.

odt2html1-Workflow unter $/workflows/ erzeugt aus einer ODT-Datei, die auf $/odt2html/templates/template1/template1.ott basiert (die hinterleg-

ten benutzerspezifischen Stilvorlagen müssen auf den Text angewendet werden), eine Ausgabe-XHTML-Datei. Die Eingabedatei kann per Parameter mitgeteilt oder per Dialog ausgewählt werden. Nach odt2html1 werden aus $/odt2html/templates/template1/ prepare4hierarchical.xsl mit xsltransformator1, html_flat2hierarchical1_config.xml mit html_flat2hierarchical1 und html_clean.xsl mit xsltransformator nacheinander aufgerufen. Das Ergebnis wird unter $/workflows/temp/output_1/output_4.html (samt den dazugehörigen Bildern, die auch in besagtem Verzeichnis in einer info.xml aufgelistet sind) abgestellt, auch werden die Zwischenergebnisse output_1.html, output_2.html und output_3.html dort nicht aufgeräumt.

odt2html2-Workflow unter $/workflows/ ruft odt2html1 auf eine Reihe von Eingabe-ODT-Dateien auf, die in einer Steuerungsdatei hinterlegt sind, und fügt diese mit html_concatenate1 zusammen. Anschließend werden aus $/odt2html/templates/template1/ prepare4hierarchical.xsl mit xsltransformator1, html_flat2hierarchical1_config.xml mit html_flat2hierarchical1 und html_clean.xsl mit xsltransformator auf die zusammengesetzte XHTML-Datei aufgerufen. Die Steuerungsdatei kann per Parameter mitgeteilt oder per Dialog ausgewählt werden, ferner ist der Pfad anzugeben, unter welchem die Ausgabedatei abgelegt werden soll. Das Ergebnis wird unter $/workflows/temp/output_1/output_4.html (im Moment noch ohne den dazugehörigen Bildern und ohne info.xml) abgestellt, auch werden die Zwischenergebnisse output_1.html, output_2.html und output_3.html sowie die output_1_*-Verzeichnisse dort nicht aufgeräumt.

odt2pdf1-Workflow unter $/workflows/ erzeugt aus einer ODT-Datei, die per Kommandozeilenargument oder per Dialog mitgeteilt werden kann, ein PDF, indem zuerst mit xsltransformator1 und /odt2html/templates/template1/prepare4html2latex1_l transformiert wird, um html2pdf1-Workflow aufrufen zu können.

odt2pdf2-Workflow unter $/workflows/ unterscheidet sich von odt2pdf1-Workflow nur dahingehend, dass per Kommandozeilenargument eine Steuerungsdatei entgegengenommen wird, die mehrere Eingabe-ODT-Dateien referenziert, welche dann mit odt2html2-Workflow für die weitere Verarbeitung vorbereitet werden. Ferner muss der Pfad angegeben werden, unter welchem die Ergebnis-PDF-Datei abgelegt werden soll.

setup1-Workflow unter $/workflows richtet die Tool-Sammlung für die Verwendung ein, indem die Dateien aus $/w3c/ in die Verzeichnisse der Tools kopiert werden, damit diese jeweils eigenständig darauf zugreifen können.

epub2wordpress1-Workflow unter $/workflows/epub2wordpress/epub2wordpress1/ entpackt ein EPUB2 mittels epub2html1 aus $/html2epub/html2epub1/, wendet das Stylesheet $/workflows/epub2wordpress/epub2wordpress1/html2wordpress1.xsl mittels xsltransformator1 aus $/xsltransformator/xsltransformator1/ darauf an und übermittelt das Ergebnis an eine WordPress-Installation mithilfe von html2wordpress1 aus $/html2wordpress/html2wordpress1/, wobei die html2wordpress1-Steuerungsdatei, die per Kommandozeilenargument mitgegeben wird, als Quelldatei $/html2wordpress/html2wordpress1/input.html eingetragen haben muss. Die in der EPUB2-Datei enthaltenen XHTML-Dateien werden jeweils einzeln transformiert und übermittelt.

odt2all1_config_edit1 unter $/workflows/gui/odt2all1_config_edit1/ dient der Workflow-Unterstützung und erlaubt das Auswählen oder Neuanlegen einer odt2all1-Workflow-Konfigurationsdatei, wobei sowohl die Eingabe-ODT-Dateien als auch die html2epub1-Konfigurationsdatei hinterlegt werden können.

odt2all1_config_edit2 unter $/workflows/gui/odt2all1_config_edit2/ un-

terscheidet sich von odt2all1_config_edit1 nur dahingehend, dass die html2epub1-Konfigurationsdatei nicht hinterlegt werden kann (weil z.B. ein Workflow sich darum kümmert).

xml_split1 unter $/xml_split/xml_split1/ verhält sich ganz wie html_split1 mit dem Unterschied, dass keine Sonderbehandlung für bestimmte HTML-Elemente wie zum Beispiel solche mit Dateipfaden erfolgt.

xsltransformator1 unter $/xsltransformator/xsltransformator1/ ist ein primitiver Wrapper, um mithilfe der Java-Standardlibrary eine XSLT-Transformation durchführen zu können, ohne ein externes Programm dafür aufrufen zu müssen. xsltransformator1 unterstützt die automatische Auflösung von DTDs noch nicht, sodass im Unterverzeichnis $/xsltransformator/xsltransformator1/entities/ die Datei config_xhtml1-strict.xml für XHTML 1.0 Strict oder config_xhtml1_1.xml für XHTML 1.0 nach config.xml kopiert werden muss, damit der Aufruf erfolgreich sein kann.

multitransform1 unter $/xsltransformator/xsltransformator1/workflows/multitransform1/ kann ein XSLT-Stylesheet zwecks XSLT-Transformation auf eine Reihe von XML-Eingabedateien anwenden, welche dem Programm per Steuerungsdatei mitgeteilt werden zusammen mit dem jeweilig gewünschten Pfad der Ausgabe-Datei. Intern wird xsltransformator1 aufgerufen.

Kapitel 4

Den Quellcode kompilieren

Weil die Tools unter der GNU Affero General Public License 3 or any later version lizenziert sind, hat jeder Nutzer der Software ein Recht darauf, den Quellcode der Programme erhalten zu dürfen und zu können, welcher verändert oder unverändert eingesetzt als auch beliebig weitergegeben werden darf, solange dies unter der gleichen Lizenz geschieht. Wenn die Tools lediglich als ausführbare Dateien (Java Bytecode) ohne Quelltext überlassen wurden oder auf einem Server ausgeführt werden, der über ein Netzwerk zugänglich ist, muss entweder kenntlich gemacht sein, wo der korrespondierende Quelltext heruntergeladen werden kann oder eine Anschrift genannt werden, die auf Anfrage den Quelltext postalisch zusendet (bis zu drei Jahre nach erfolgter binärer Distribution). Wenn es sich um die offiziellen Bezugsquellen wie das öffentliche Online-Repository https://github.com/publishing-systems/automated_digital_publishing[1] oder die Download-Packages unter http://www.publishing-systems.org/downloads.php[2] handelt, ist sichergestellt, dass der Quellcode mit Java 1.6 und höher sowie mit OpenJDK[3] kompatibel ist. Die Kompilierbarkeit soll für die 100% freien Distributionen des GNU-Betriebssystems[4] gewährleistet werden, wobei insbesondere gNewSense[5] als Referenzsystem gelten kann. Auf debianbasierten Distributionen kann mit

apt-get install openjdk-6-jdk

ein JDK installiert werden. Weil GNU make üblicherweise vorinstalliert ist, genügt der Befehl

make

im Wurzelverzeichnis $/.

Da das offizielle öffentliche Repository auf GitHub gehostet wird, ist kollaboratives Arbeiten an der Software besonders einfach. Mit einem GitHub-Account kann zunächst ein „Fork“, eine eigene Kopie des Repositorys angelegt werden. Sofern git mittels

apt-get install git

installiert wurde, kann mit dem Befehl

git clone <HTTPS clone URL des Repositorys auf github.com>

eine lokale Kopie heruntergeladen werden, in welcher die Entwicklung statt-

[1] `https://github.com/publishing-systems/automated_digital_publishing`
[2] `http://www.publishing-systems.org/downloads.php`
[3] `http://openjdk.java.net/`
[4] `https://www.gnu.org/distros/free-distros.html`
[5] `http://www.gnewsense.org/`

findet. Ist die Programmierung abgeschlossen, können die Änderungen mit dem Dreisatz

```
git add *
git commit -m "Beschreibung der Änderungen"
git push origin <Branch, erst mal master>
```

in den eigenen Fork hochgeladen und damit veröffentlicht werden. Zu berücksichtigen ist dabei, dass alle Änderungen ebenfalls unter GNU Affero General Public License 3 or any later version lizenziert werden müssen. Auf github.com kann dann in der Verwaltung des Fork-Repositorys ein „Pull Request“ zwecks potentieller Übernahme in das offizielle Repository angelegt werden, welcher nach Prüfung angenommen oder abgelehnt wird. Auf diese Weise können Patches zur Korrektur von Fehlern, neue Features, neue Tools usw. eingereicht werden, oder aber die Werkzeugkiste auch in eine ganz andere Richtung weiterentwickelt werden.

Kapitel 5

Lizenzbestimmungen

5.1 GNU AFFERO GENERAL PUBLIC LICENSE

5.2 Preamble

The GNU Affero General Public License is a free, copyleft license for software and other kinds of works, specifically designed to ensure cooperation with the community in the case of network server software.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, our General Public Licenses are intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

Developers that use our General Public Licenses protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License which gives you legal permission to copy, distribute and/or modify the software.

A secondary benefit of defending all users' freedom is that improvements made in alternate versions of the program, if they receive widespread use, become available for other developers to incorporate. Many developers of free software are heartened and encouraged by the resulting cooperation. However,

[1] `http://fsf.org/`

in the case of software used on network servers, this result may fail to come about. The GNU General Public License permits making a modified version and letting the public access it on a server without ever releasing its source code to the public.

The GNU Affero General Public License is designed specifically to ensure that, in such cases, the modified source code becomes available to the community. It requires the operator of a network server to provide the source code of the modified version running there to the users of that server. Therefore, public use of a modified version, on a publicly accessible server, gives the public access to the source code of the modified version.

An older license, called the Affero General Public License and published by Affero, was designed to accomplish similar goals. This is a different license, not a version of the Affero GPL, but Affero has released a new version of the Affero GPL which permits relicensing under this license.

The precise terms and conditions for copying, distribution and modification follow.

5.3 TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU Affero General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of

your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces

that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the

work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law.

If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void,

and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you

entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Remote Network Interaction; Use with the GNU General Public License.

Notwithstanding any other provision of this License, if you modify the Program, your modified version must prominently offer all users interacting with it remotely through a computer network (if your version supports such interaction) an opportunity to receive the Corresponding Source of your version by providing access to the Corresponding Source from a network server at no charge, through some standard or customary means of facilitating copying of software. This Corresponding Source shall include the Corresponding Source for any work covered by version 3 of the GNU General Public License that is incorporated pursuant to the following paragraph.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the work with which it is combined will remain governed by version 3 of the GNU General Public License.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU Affero General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU Affero General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU Affero General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU Affero General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

5.4 How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU Affero General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Affero General Public License for more details.

You should have received a copy of the GNU Affero General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If your software can interact with users remotely through a computer network, you should also make sure that it provides a way for users to get its source. For example, if your program is a web application, its interface could display a "Source" link that leads users to an archive of the code. There are many ways you could offer source, and different solutions will be better for different programs; see section 13 for the specific requirements.

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU AGPL, see <http://www.gnu.org/licenses/[2]>.

[2] `http://www.gnu.org/licenses/`