Universität Leipzig
Wirtschaftswissenschaftliche Fakultät
Institut für Wirtschaftsinformatik
Professur Informationsmanagement

Thema:

Integration von Formatting-Object-Transformern zur automatischen Generierung von PDF-Dokumenten unter besonderer Berücksichtigung der Ausgabe arabischer und gemischtsprachiger Texte mit unterschiedlichen Schriftausrichtungen

Diplomarbeit

zur Erlangung des akademischen Grades

Diplom-Wirtschaftsinformatiker (Dipl.-Wirtsch.-Inf.)

Betreuender Hochschullehrer: Prof. Dr. phil. habil. E. Schulz
Technischer Betreuer: Dipl.-Inf. J. Kupferschmidt
Bearbeiter: Abdessamed Karmoun
Klasingstr. 9
04315 Leipzig
Matr.-Nr.: 8956981

Eingereicht am: 03.06.2010

Danksagung

Nach der Fertigstellung meiner Diplomarbeit möchte ich allen danken, die mir bei der Arbeit direkt oder indirekt geholfen haben. Insbesondre möchte ich danken:

Meinen universitäts- und arbeitsseitigen Diplomarbeitsbetreuern, Prof. Dr. E. Schulz und Dipl.-Inf. J. Kupferschmidt für das wertvolle Fördern.

Meinen Eltern, meiner Frau, meinen Kindern, Geschwistern und Freunden für die finanzielle, moralische und psychische Unterstützung.

Für die organisatorische Betreuung, Dr. T. Hanstein.

Für eine außergewöhnlich gute Arbeitsatmosphäre, allen Kollegen des Universitätsrechenzentrums; besonders in der Abteilung des Rechnerbetriebes .

Abstract

Thema:	Integration von Formatting-Object-Transformern zur automatischen Generierung von PDF-Dokumenten unter besonderer Berücksichtigung der Ausgabe arabischer und gemischtsprachiger Texte mit unterschiedlichen Schriftausrichtungen.
Diplomand:	Abdessamad Karmoun Klasingstr. 9 04315 Leipzig
Betriebsort:	Universitätsrechenzentrum Ritterstr. 30-36 04109 Leipzig
Betreuender Hochschullehrer:	Prof. Dr. phil. habil. E. Schulz
Technischer Betreuer:	Dipl.-Inf. J. Kupferschmidt
Abgabedatum:	03.06.2010
Schlagworte:	PDF, Arabisch, XML, XSL-FO, Standard, Unicode, Bidirektionalität, FO- Prozessor, Antenna House XSL Formatter, RenderX XEP, Apache FOP, Darstellung.

Im Rahmen einer Reihe von Projekten zur datenbankgestützten Erschließung und digitalen Bereitstellung der orientalischen Handschriften sowohl der Universitätsbibliothek Leipzig als auch von verschiedenen Institutionen in Indonesien, entstand der Bedarf, die in den Datenbanken vorhandenen Informationen dem Benutzer der Webanwendung in einer bequemen und barrierefreien Form zur Verfügung zu stellen.
PDF stellt ein wichtiges Instrument zur Darstellung, Übertragung und Archivierung elektronischer Dokumente dar. Bei der automatischen Erstellung von PDF-Dokumenten entstanden immer Probleme bei der Darstellung arabischer Schrift.
Das Ziel der Diplomarbeit ist auf der einen Seite die systematische Beschreibung der Probleme der Darstellung arabischer Schrift und auf der anderen Seite die Untersuchung, Evaluierung und Integration von Formatierern in das Projekt, die eine automatische Generierung von PDF-Dokumenten unter besonderer Berücksichtigung der Ausgabe arabischer und gemischtsprachiger Texte mit unterschiedlichen Schriftausrichtungen ermöglichen.
Diese Projekte sind auf dem CMS MyCoRe basiert, und von der Deutschen Forschungsgemeinschaft (DFG) und vom Auswärtigen Amt der Bundesrepublik Deutschland finanziell unterstützt, und durch das Universitätsrechenzentrum Leipzig technisch realisiert.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AHF	Antenna House Formatierer
FOP	Formatting Objects Processor
JVM	Java Virtual Machine
HTML	Hypertext Markup Language
PDF	Portable Document Format
TIFF	Tagged Image File Format
UTF8	Unicode-Format
W3C	Word Wide Web Consortium
XEP	XEP Rendering Engine
XML	Extensible Markup Language
XRF	Xmlroff
XSL	Extensible Stylesheet Language
XSL-FO	Extensible Stylesheet Language – Formatting Objects
XSLT	Extensible Stylesheet Language – Transformations

1 Einleitung

1.1 Vorwort

Diese Diplomarbeit wurde im Rahmen einer Reihe von Projekten zur datenbankgestützten Erschließung und digitalen Bereitstellung der arabischen, persischen, türkischen und indonesischen Handschriften sowohl der Universitätsbibliothek Leipzig als auch von verschiedenen Institutionen in Indonesien geschrieben[17][18].

Diese Projekte sind von der Deutschen Forschungsgemeinschaft (DFG) und vom Auswärtigen Amt der Bundesrepublik Deutschland gefördert und durch das Universitätsrechenzentrum Leipzig technisch realisiert worden.

Für die technische Umsetzung wird das CMS MyCoRe eingesetzt[19], welches von verschiedenen deutschen Universitäten (u.a. URZ der Universität Leipzig), die einen Open Source Kern zur Unterstützung bei Aufgaben aus den Bereichen digitale Bibliotheken und wissenschaftliche Sammlungen bereitstellen, entwickelt wird.

Das Ziel der Diplomarbeit ist auf der einen Seite die systematische und technische Beschreibung der Problematik der Darstellung arabischer Schrift und auf der anderen Seite die Untersuchung, Evaluierung und Integration von Formatierern in das Projekt, die eine automatische Generierung von PDF-Dokumenten unter besonderer Berücksichtigung der Ausgabe arabischer und gemischtsprachiger Texte mit unterschiedlichen Schriftausrichtungen ermöglichen.

Die erzeugten PDF-Dokumente können Daten, Schriftsätze, Bilder usw. beinhalten.

Die Daten werden in den XML-Dokumenten gespeichert. Aus diesen XML-Dokumenten werden die Daten für die PDF-Dokumente ausgelesen. Die Darstellung der Daten erfolgt durch den Einsatz von XSL. XSL-FO wird für die Gestaltung von PDF-Dokumenten eingesetzt.

Für die Formatierung der Daten in PDF-Dokumente werden die verschiedenen Formatierer FOP, Xmlroff, AHF und XEP untersucht, evaluiert und eingesetzt.

1.2 Problembeschreibung

Das World Wide Web ist ein spezieller Dienst des Internets, das jedem Benutzer durch eine einfache Oberfläche erlaubt, gezielt nach Informationen zu suchen. Diese Informationen aus dem Internet unterstützen uns in vielen Bereichen des Lebens, besonders in der Wissenschaft, Wirtschaft und Verwaltung. Aber die Benutzer des Webs müssen nicht nur die Informationen finden und auf dem Bildschirm sehen, sondern sie auch in einer druckbaren Form zur Verfügung zu haben. Aus diesem Grund sollten die Entwickler die Informationen nicht nur im Browser darstellen, sondern auch den Benutzern ermöglichen, diese Informationen richtig, leicht und barrierefrei auszudrucken. Das ist besonders wichtig für solche Projekte, die sich mit dem Dokumentieren und Archivieren von heterogenen und gemischten Inhalten beschäftigen.

Die korrekte Darstellung und Übertragung von arabischen und gemischtsprachigen Texten mit unterschiedlichen Schriftausrichtungen stellt heutzutage eine große Herausforderung für die IT-Entwickler dar, weil die arabische Sprache eine ganz andere Logik beim Schreiben enthält wie die Right to Left-Ausrichtung, Ligatur, künstliche Fonts usw.. Zur Bewältigung solcher Problematiken wurden mittlerweile verschiedene Techniken und Werkzeugen entwickelt.

Portable Document Format (PDF) stellt ein wichtiges Instrument zum Ausdrucken, Übertragen und Archivieren von Daten in einer korrekten Form dar. PDF ist ein plattformunabhängiges Dateiformat für elektronische Dokumente, das es ermöglicht, Texte, Bilder, Grafiken und auch Fonts in einem bestimmten Format zu speichern, so dass sie genauso dargestellt werden, wie sie in PDF gespeichert wurden. D.h. die Darstellung der Informationen beim Benutzer ist unabhängig von den bei ihm vorhandenen Fonts. Neben PDF gibt es noch eine Reihe von Techniken, die in den nächsten Kapiteln diskutiert werden.

1.3 Vorgehensweise

In den auf MyCoRe basierten Projekten wurde strikt zwischen Daten, Logik und Darstellung getrennt. Als Grundlage des Speicher- und Datenaustauschformates wird MyCoRe XML verwendet. Die Benutzerschnittstelle einer Anwendung wird durch Java Servlets implementiert, die dem Benutzer interaktive Abläufe und Funktionen zur Verfügung stellen. Die XML-Darstellung wird mittels XSL-Stylesheets in eine Webanwendung im HTML-Format transformiert. Die Verwendung von XSL ermöglicht MyCoRe sowohl unterschiedliche Darstellungen zu erzeugen, als auch mehrere XML-Dokumente miteinander zu kombinieren. Die Darstellungsebene in MyCoRe erstreckt sich von einfachen Webseiten über Trefferlisten und Resultatanzeigen bis hin zu Komponenten der Benutzer- und Datenverwaltung[19].
Die Vorgehensweise der PDF-Erstellung erfolgt zusammengefasst so, dass man ein XML Dokument und ein XSL Stylesheet hat, das ein XSLT Prozessor automatisch in eine FO-Datei transformiert. Danach werden diese FO-Datei und die dazugehörigen Grafiken mit Hilfe eines XSL-Formatierers verwendet, um ein PDF zu generieren.

2 Ausgabe arabischer und gemischtsprachiger Texte mit unterschiedlichen Schriftausrichtungen

2.1 Einführung in die arabische Sprache

2.1.1 Allgemeines

Arabisch ist die Sprache einer reichhaltigen Kultur, die eine jahrhundertelange Tradition hat. Die arabische Sprache ist eine bemerkenswert analytische und mathematische Sprache. Sie ist eine der reichsten Sprachen der Welt, in der jeder Buchstabe einen mathematischen Wert hat und alle arabischen Wörter sich mit wenigen Ausnahmen(Fremd- und Lehnwörtern) auf drei Konsonanten, die sogenannte Wurzel, reduzieren lassen. Darum ist sie eher geeignet, um wissenschaftlich-hypothetische und philosophische Gedanken durch eine überzeugend flexible Grammatik und Syntax nahezu verlustfrei zu formulieren[2]. Weiterhin ist die arabische Sprache die verbreiteteste Sprache des semitischen Zweigs der afroasiatischen Sprachfamilie. Schätzungen gehen davon aus, dass Arabisch von 240 Millionen Menschen als Muttersprache und von weiteren 50 Millionen als Zweitsprache gesprochen wird. Hinzu ist sie die Sprache des Islams, dem mehr als eine Milliarde Menschen angehören und ist somit die verbindende Sprache der Muslime. Deshalb umfasst die arabische Sprache eine Vielzahl von verschiedenen Sprachformen. Was all diese Sprachformen zu einer Sprache zusammenbindet, ist vor allem der Koran. Die moderne arabische Standardsprache beruht auf dem klassischen Arabischen, der Sprache des Korans, und unterscheidet sich von den gesprochenen Varianten der arabischen Dialekte. Die einzelnen arabischen Dialekte in

den verschiedenen Ländern unterscheiden sich zum Teil sehr stark voneinander, und oftmals ist die Kommunikation zwischen arabischen Muttersprachlern unterschiedlicher Herkunft in ihren Dialekten recht schwierig. Um diese Schwierigkeit zu überbrücken, benutzen viele Sprecher das Standardarabisch als multifunktionale Hochsprache.

2.1.2 Arabische Schrift

Die arabische Schrift ist eine ganze normale Kurrentschrift, die heute eine der wichtigsten Schriften der Welt ist. Sie hat von den semitischen Schriften die größte Verbreitung erlangt. Mit dem Islam und dem Koran wurde sie von Volk zu Volk getragen. So findet man die arabische Schrift im Gebrauch nicht nur zur Wiedergabe der arabischen Sprache, sondern auch für die persische Sprache, die kurdische Sprache (in Irak, Iran und Syrien), Türkisch und (früher) Tatarisch, die malaysische Sprache und für Paschtu und Urdu, ja sogar für Somali (vgl. Wadaad-Schrift), Swahili und Hausa sowie für einige Berbersprachen (selten). Solche Verwendungen des arabischen Alphabets für andere Sprachen werden als ʿağamī-Schrift bezeichnet.

Im Laufe der Geschichte hat die arabische Schrift sich verändert, da immer mehr Buchstaben in der Gestalt zusammenfielen. Diese werden durch Punkte über und unter den alten Konsonanten unterschieden. Ohne die Punkte fallen beispielsweise in der beidseitig verbundenen Form die Buchstaben N, T, TH, B, Y und P zusammen: (nacheinender: a-l-N-T-Th-B-Y-P-a: النتثبيا) Die Punkte für das P übernahm man aus dem Persischen, um Fremdwörter, die ein P enthalten, wiederzugeben. In einer früheren Form der arabischen Schrift, dem Kūfī (كوفى), in der es noch keine Punkte gab, wurden viele Texte fast nicht mehr lesbar, da wie bereits erwähnt nur die Konsonanten geschrieben wurden und einige davon auch nicht mehr zu unterscheiden waren. Die neue entwickelte Schrift mit den Punkten nennt man Nasḫī (نسخ) (siehe Abbildung 1).

Abbildung 1: Einige Formen der arabischen Schrift[25][26][27][28][29][30]

2.1.3 Besonderheiten der arabischen Sprache

2.1.3.1 Right to Left

Das arabische Alphabet besteht aus 28 Buchstaben. Diese werden von rechts nach links geschrieben. Dabei werden mit sechs Ausnahmen alle Buchstaben entsprechend der Laufrichtung der arabischen Schrift von rechts nach links verbunden. Außerdem sind die Buchstaben unterschiedlich groß, es gibt jedoch keine Groß- und Kleinschreibung (siehe Abbildung 2). [4]

i	ii	iii	iv	v	vi	vii	i	ii	iii	iv	v	vi	vii
1.	1	ا	ـا			أَلِفٌ	15.	800	ض	ـض	ـضـ	ضـ	ضَادٌ
2.	2	ب	ـب	ـبـ	بـ	بَاءٌ	16.	9	ط	ـط	ـطـ	طـ	طَاءٌ
3.	400	ت	ـت	ـتـ	تـ	تَاءٌ	17.	900	ظ	ـظ	ـظـ	ظـ	ظَاءٌ
4.	500	ث	ـث	ـثـ	ثـ	ثَاءٌ	18.	70	ع	ـع	ـعـ	عـ	عَيْنٌ
5.	3	ج	ـج	ـجـ	جـ	جِيمٌ	19.	1000	غ	ـغ	ـغـ	غـ	غَيْنٌ
6.	8	ح	ـح	ـحـ	حـ	حَاءٌ	20.	80	ف	ـف	ـفـ	فـ	فَاءٌ
7.	600	خ	ـخ	ـخـ	خـ	خَاءٌ	21.	100	ق	ـق	ـقـ	قـ	قَافٌ
8.	4	د	ـد			دَالٌ	22.	20	ك	ـك	ـكـ	كـ	كَافٌ
9.	700	ذ	ـذ			ذَالٌ	23.	30	ل	ـل	ـلـ	لـ	لَامٌ
10.	200	ر	ـر			رَاءٌ	24.	40	م	ـم	ـمـ	مـ	مِيمٌ
11.	7	ز	ـز			زَايٌ	25.	50	ن	ـن	ـنـ	نـ	نُونٌ
12.	60	س	ـس	ـسـ	سـ	سِينٌ	26.	5	ه	ـه	ـهـ	هـ	هَاءٌ
13.	300	ش	ـش	ـشـ	شـ	شِينٌ	27.	6	و	ـو			وَاوٌ
14.	90	ص	ـص	ـصـ	صـ	صَادٌ	28.	10	ي	ـي	ـيـ	يـ	يَاءٌ

Abbildung 2: Das arabische Alphabet. Legende: i) Nummer – ii) Zahlwert – iii) isolierte Form – iv) nach rechts verbundene Form – v) beidseitig verbundene Form – vi) nach links verbundene Form – vii) Name[22]

2.1.3.2 Bidirektionaler Text

In verschiedenen Sprachen werden unterschiedliche Schreibsysteme verwendet. Zum Beispiel: In Europa und in europäisch beeinflussten Kulturen sind hauptsächlich Schriften mit einer Schreibrichtung von links nach rechts gewöhnlich, beispielsweise in Latein, Deutsch oder Englisch. Im Gegenteil dazu wird in den semitischen Schriften beispielsweise Arabisch von rechts nach links geschrieben. Während die arabischen Buchstaben von rechts nach links geschrieben werden, schreibt man die Zahlen von links nach rechts, wodurch ein Wechsel in der Schreibrichtung erfolgt. Das Gleiche geschieht auch beim Einfügen von Wörtern aus lateinischen Buchstaben, die ja ebenfalls von links nach rechts geschrieben werden müssen. Stehen arabische Wörter gemeinsam mit solchen aus lateinischen Buchstaben gebildeten Wörtern oder auch mit Zahlen auf einer Zeile, so spricht man von einem bidirektionalen Text, da die Schreibrichtung wechselt.

2.1.3.3 Ligatur

Ligatur ist in der Typografie die Verschmelzung zweier oder mehrerer Buchstaben zu einer optischen und formalen Einheit, um optische Lücken, die beim schnellen Lesen stören, zu vermeiden. Eine weitere Besonderheit des Arabischen besteht in der Existenz von Ligaturen. Dabei werden verschiedene Buchstaben zu einer Drucktype verbunden (siehe Abbildung 3).

Abbildung 3: Form des Namens Muhammad, oben als Ligatur, unten auf der Grundlinie verbunden, wie in einfacheren Drucken üblich[23]

2.1.3.4 ḥarakāt

Die arabische Schrift besteht aus 28 Buchstaben, von denen drei lange Vokale darstellen. Zusätzlich zu den Konsonanten und langen Vokalen des arabischen Alphabets gibt es

noch viele andere Laute, welche in der Regel nicht geschrieben werden. Diese nennen sich ḥarakāt oder Vokalzeichen. Ohne diese Vokalzeichen kommt der Lernende niemals klar. Sie sind eine große Hilfe, um richtig lesen zu können. Besonders am Anfang sind sie unverzichtbar. Diese werden aber nicht als Bestandteil des Alphabets angesehen, sondern als optionale Erweiterung des arabischen Alphabetes. Sie werden zur Vokalisation von Texten verwendet. Dabei werden sie durch kurze Zeichen ersetzt, die sich über oder unter den Buchstaben sein sollen, damit man die korrekte Aussprache des Wortes hat. Zum Beispiel: das kasra unter den davor gesprochenen Buchstaben gesetzt. Das sukūn zeigt an, dass nach dem Buchstaben kein Vokal folgt (siehe Abbildung 4).

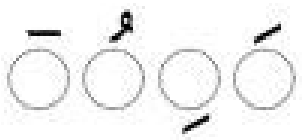


Abbildung 4: ḥarakāt[24]

Die ḥarakāt werden meist nur in Urkunden oder religiösen Texten geschrieben. Da die Wörter in arabischen Ländern bekannt sind, wird auf solche Zeichen meist verzichtet sowie im täglichen Leben nie benutzt. Deshalb sind Texte in Tageszeitungen beispielsweise nicht vokalisiert, d.h. die ḥarakāt werden nicht geschrieben. Den Sprachanfängern fällt es bei solchen Texten ohne Vokalzeichen schwer, die richtige Aussprache zu treffen, weshalb die arabische Sprache ohne diese Zeichen wie eine unlösbare Aufgabe wirkt.

2.2 Probleme bei der Darstellung arabischer Schrift

2.2.1 Problem der Right-to-Left-Schriftrichtung

Das Schreiben von rechts nach links ist auch nach einer lebenslangen Gewöhnung der "links-rechts" Schreibrichtung nicht einfach zu bewerkstelligen. Auch die so genannten Verbindungen der einzelnen Buchstaben erschweren die Schrift. Die Formatierung Rechts-Links stellt vor allem in der Informations- und Computertechnik ein Problem dar. Dafür braucht man gut geeignete Programme, die Schreibrichtungen von rechts nach links unterstützen. Durch die Aktivierung der entsprechenden Einstellung wird es

für den Benutzer möglich, arabische Texte einzugeben, zu ändern und automatisch mit einer Absatzausrichtung und Textrichtung von rechts nach links anzuzeigen. Der Bildschirm soll auch von rechts nach links aufgebaut werden.

2.2.2 Problem der Bidirektionalität

Die Bidirektionalität stellt vor allem in der Informations- und Computertechnik, seit dem weltweiten Datenaustausch durch das Internet, ein Problem dar. Damit haben bis heute manche Computerprogramme Darstellungsprobleme. Sie können die arabische Schrift nicht richtig verfassen und anzeigen, zum Beispiel funktioniert die PDF-Erstellung manchmal nicht richtig. Die bidirektionalen Texte erscheinen daher ebenfalls als Buchtstabe für Buchstabe von links nach rechts geschrieben, was natürlich unsinnig ist. Das hier nur kurz geschilderte Problem stellt besondere Anforderungen an ein Computersystem, um Arabisch zu verfassen und wiedergeben zu können. Es muss in der Lage sein, bidirektionalen Text darzustellen, eine Kontextanalyse durchzuführen und es sollte mit diakritischen Zeichen umgehen können. Dafür gibt es aber verschiedene Algorithmen, welche versuchen, eine passende Schreibrichtung für die Satzzeichen zu ermitteln.

2.2.3 Problem der Buchstabenverbindung

Da die arabische Schrift grundsätzlich nur wortweise verbunden, geschrieben und gedruckt wird, ist die Form des Zeichens vom Kontext abhängig. Diese verändert sich in Abhängigkeit von der Stellung der Buchstaben im Wort. Deshalb gibt es verschiedene Varianten, je nachdem, ob der Buchstabe in Anfangs-, Mittel-, bzw. Endposition oder isoliert steht. Welche Form ein Buchstabe schließlich annimmt, hängt von seinem rechten und linken Nachbarn ab. Es gibt bis zu vier verschiedene Formen je Buchstabe: allein stehend, initial, medial und final. Die einzige Zwangsligatur im Arabischen ist das lām-ʾalif (ﻻ) : Es entsteht bei der Verbindung der Buchstaben lām (ل)und ʾalif (ا).

Die Komplexität des Schreibens entsteht durch die Verbindungsmöglichkeiten der Buchstaben zueinander, die dann das Fließende, das Kalligrafische ergeben. Erschwe-

rend kommt noch dazu, dass nicht alle Buchstaben miteinander verbunden werden. Weiterhin haben die Buchstaben, die verbunden werden müssen, je nach Stellung im Wort (Anfang-Mitte-Ende) verschiede Formen (siehe Abbildung 2). [3]

2.2.4 Problem der arabischen Fonts

Die Islamische Kultur hat eine enge Verbindung zwischen Sprache, Schrift und Religion aufgebaut. Diese Verbindung wird auch in der Schrift reflektiert. Dadurch findet man einen Einstieg in die Kalligrafie. Dieser Aspekt der islamischen Kunst hat sich aus der arabischen Schrift in engem Zusammenhang mit dem Islam entwickelt, was dazu führt, dass die Araber an die Schrift hohe ästhetische Anforderungen stellen. Somit sind kalligrafische Fertigkeiten auch für jede arabische Werbeagentur unverzichtbar, da jede Werbung, die nicht kunstvoll mit der Schrift umgeht, an den Ansprüchen des Marktes vorbeigeht. Ich werde mich auf sieben Schriftarten beschränken, wenn auch die Literatur teilweise zwischen 10 bis 12 verschiedenen Schriftausprägungen differenziert. Allerdings stellen die von mir aufgezeigten sieben Arten die Hauptstile dar. [5]

Kūfī: Die kufische Schrift ist eine der ältesten Formen der arabischen Schrift und gehört zu den Konsonantenschriften. Sie wurde erst kurz vor dem Islam bei den Arabern eingeführt. Die kufische Schrift ist nach der Stadt Kufa benannt. Dort wurde sie entwickelt. Obwohl sie zum Beginn eine Monumentalschrift war, die steil und geometrisch verlief, wurde sie mit dem Aufkommen des Islam die dominante Schrift für den Koran und im religiösen Bereich in der Frühzeit des Islams (siehe Abbildung 5).

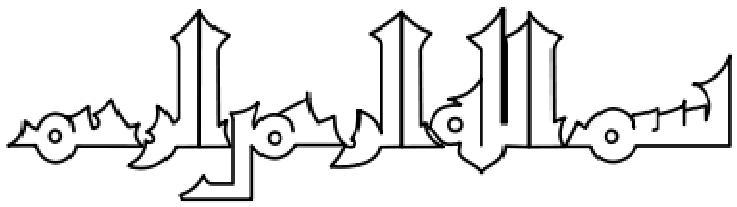


Abbildung 5: Zeichnung einer Inschriften-basmala im kufischen Duktus, 9. Jahrhundert. Das Original befindet sich im Islamischen Museum in Kairo (Inventar-Nr. 7853)[25]

Nasḫī: Die Nasḫī-Schrift „abschreiben, kopieren“ war eine der frühesten Schriftarten und nach einer Reform durch Ibn Mulqlah im 10. Jahrhundert wurde sie zu einer der beliebtesten Schriftarten. Die Nasḫī-Schrift hat sich im Alltag beim Druck durchgesetzt und ist allgemein die Bezeichnung für alle arabischen Kursiven, die zum Kopieren von Büchern verwendet werden. Aufgrund ihrer Eleganz und ihrer besseren Lesbarkeit ersetzte sie allmählich die Kūfī-Schrift als Hauptschrift für das Schreiben des Korans, und aus diesen beiden Schriften entwickelten sich im Laufe der Zeit in verschiedenen Gebieten zahlreiche Varianten (siehe Abbildung 6).

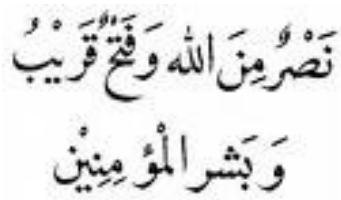


Abbildung 6: Die Nasḫī-Schrift[26]

Ṯuluṯ: Die Ṯuluṯ-Schrift „Drittel“ zeichnet sich durch besondere Biegsamkeit der Buchstaben aus und findet sich oft als Zierschrift an Moscheen und anderen Gebäuden. Dieser Typus entstand schon im 7. Jahrhundert - war aber erst im späten 9. Jahrhundert voll entwickelt. Sie wird nicht so sehr für Koranniederschriften gebraucht, sondern mehr für kalligrafische Inschriften (siehe Abbildung 7).

Abbildung 7: Ṯuluṯ von Mehmed Izzet Efendi (1841–1904)[27]

Req’a: Die Req'a-Schrift ist eine Entwicklung aus der Ṯuluṯ-Schrift, die heute die bevorzugte Schrift in den arabischen Ländern ist. Diese Schriftart wird meist für Handschriften verwendet (siehe Abbildung 8).

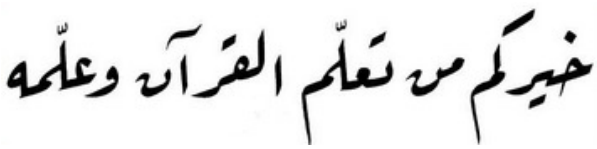


Abbildung 8: Req´a –Schrift[28]

Taʿlīq: Die Taʿlīq-Schrift ist der Farsi Schriftstil. Er wurde in Persien während des 13. Jahrhunderts entwickelt, wobei Elemente aus Nasḫī, Tauqīʿ und Req`a verwendet wurden. Diese Schrift wurde hauptsächlich eingesetzt, um Bücher und Diwane niederzuschreiben (siehe Abbildung 9).

Abbildung 9: Taʿlīq-Schrift[29]

Nastaʿlīq: Die Nastaʿlīq-Schrift ist eine besondere Stilart der persischen Kalligrafie. Sie ist eine Mischung aus Nasḫī und Taʿlīq und wurde von Mir Ali Tabrizi, einem persischen Kalligrafen aus dem 14./15. Jhd. in Täbris, geschaffen. Im 15. Jahrhundert wurde sie die am meisten gebrauchte Schrift Persiens und verbreitete sich von da aus nach Osten. Leider ist das Nastaʿlīq entsprechend seiner kalligrafischen Natur schwer als Font für die Computerdarstellung zu fassen (siehe Abbildung 10).

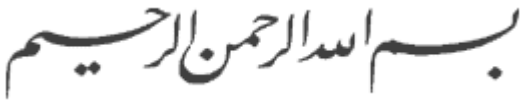


Abbildung 10: Nastaʿlīq-Schrift[30]

Dīwānī: Die Dīwānī-Schrift wurde im ausgehenden 15. Jahrhundert für Erlasse des Sultans und für die Administration des osmanischen Reiches entwickelt, um die Fälschung offizieller Dokumente zu erschweren. Deshalb ist ihr Stil sehr kompliziert.

Die Anwendung dieser Schriftart war in den Kanzleien des Osmanischen Reiches üblich. (siehe Abbildung 11).

Abbildung 11: Dīwānī-Schrift von Mehmed Izzet Efendi[31]

Damit die PDF-Erstellung ganz korrekt funktioniert, muss man die nötigen arabischen Fonts zusammentragen. Ansonsten kann man die wiedergegebenen PDFs nicht lesen, weil die entsprechenden Fonts nicht unterstützt sind.

2.3 Lösungsmöglichkeiten

2.3.1 Portable Document Format

PDF ist ein plattformunabhängiges Dateiformat für elektronische Dokumente. Dieses kann unabhängig vom ursprünglichen Anwendungsprogramm, vom Betriebssystem oder von der Hardware-Plattform originalgetreu durch das Internet überall auf der Welt verbreitet werden. Neben Text, Bildern und Grafik unterstützt es eine Funktion für die mehrsprachige Formatierung. Dadurch kann man Unternehmensbroschüren und sonstige Dokumente in allen gewünschten Sprachen in einer einzelnen PDF-Datei bereitstellen, die auf jedem Computer angezeigt und ausgedruckt werden können. Ein mehrsprachiges PDF-Dokument hat ein geringeres Volumen als mehrere Einzeldokumente in den jeweiligen Sprachen. Neben der Platzeinsparung auf Festplatten kann das mehrsprachige PDF durch die elektronischen Medien schnell verbreitet werden wie z.B. CD-ROM, Diskette oder Internet. Dazu vereinfachen die CDs oder DVDs die Verwaltung solcher Daten. Im Gegensatz dazu kann ein Dokumentpapier nicht so schnell wie seine emulierte elektronische Version überall in die Welt übermittelt werden. [1]

2.3.2 Unicode

Unicode ist ein internationaler Standard. Dabei wird für jedes sinntragende Schriftzeichen oder Textelement aller bekannten Schriftarten und Zeichensysteme ein langfristig digitaler Code festgelegt. Dies erlaubt uns, alle Buchstaben, die wir benötigen, in eindeutiger Weise zu schreiben. Dadurch kann man Probleme bei der Verwendung unterschiedlicher und inkompatibler Kodierungen in verschiedenen Ländern oder Kulturkreisen beseitigen sowie das umständliche Umcodieren beim Datenaustausch vermeiden.

Die Unicode-Kodierung liefert einen kodierten Zeichensatz für Skripte, Ziffern, Zeichen, und Buchstaben für die meisten Sprachen der Welt. Dazu bietet es andere Spezifikationen, wie z.B. die Unicode-Datenbank: Sie zeigt die Schreibrichtung für jeden Buchstaben sowie andere Informationen, die Zeilenumbrucheigenschaften: Sie beschreiben die Eigenschaften von jedem Buchstabe, welche ein Abbruch vor oder nach dem Buchstabe ermöglichen oder verhindern und der bidirektionale Algorithmus: Er legt fest, wie mehrdeutige Zeichen zwischen Texten mit verschiedener Schriftrichtung (links-rechts versus rechts-links) verarbeitet werden und wie im Fall von gemischten Texten zu verfahren ist. Dies deckt jene Probleme ab, welche bei Zitaten (Deutsch mit eingebettetem Arabisch) auftreten können. Diese Spezifikationen sind die Grundlagen für die Entwicklung von Software, die mehrsprachige Dokumente mit bidirektionalen Texten verarbeiten können. [1]

2.3.3 XML

XML ist eine Auszeichnungssprache zur Darstellung von hierarchisch strukturierten Datensätzen in Form von Textdaten und sie bezeichnet ein universelles Datenformat für die Publikation und den Plattform- und Anwendungsprogramm unabhängigen Austausch von strukturierten Dokumenten im Internet oder in Intranets. Deshalb ist es nötig, die Daten in XML-Dateien zu speichern.

2.3.4 XSL

XSL ist eine Ergänzungssprache zu XML, um Layouts für XML-Dokumente zu definieren. Sie wurde entworfen, um mehrsprachige Dokumente bei der Formatierung zu unterstützen. Dabei wird eine Menge von Layoutobjekten definiert, die die Eigenschaften von den geschriebenen mehrsprachigen Texten beschreibt und darstellt. Durch die XSL-FO Erweiterung kann man arabische Textteile von rechts nach links darstellen. [1]

3 Techniken und Werkzeuge zur PDF-Dokumentenerstellung

3.1 Techniken

3.1.1 XML mit UTF8

XML-Technologie verwendet UTF8 zur Kodierung von Unicode-Zeichen. UTF8 ist eine Abkürzung für Unicode Transformation Format 8-bit und ist die am weitesten verbreitete internationale Kodierung für Unicode-Zeichen auf Basis der ISO/IEC-10646-Norm mit mindestens 8 Bit Zeichenbreite. Dabei wird jedem Zeichen eine extra kodierte Kette von eins bis zu vier Byte von variabler Länge zugeordnet. UTF8 spielt heutzutage eine zentrale Rolle als globale Zeichenkodierung im Internet[32].

Die lokale Kodierung jedes Landes kann auch mit XML-Dokumenten spezifiziert werden. Eine XML-Datei kann in mehrere Dateien aufgeteilt werden. Die Grafiken können mit externen Dateien verknüpft werden. Das ist sehr praktisch und auf diese Weise kann man auch die Textteile in anderen Sprachen in getrennten Dateien speichern. Dazu werden die mehrsprachigen Grafiken als gemeinsame Dateien benutzt. Schließlich können alle Dateien zusammen verknüpft werden, um ein komplettes Dokument zu erstellen. Nun brauchen wir einen Formatierer, der diese Dateien auffassen sowie verarbeiten und formatieren kann, damit sie weitergegeben werden können. Dafür ist die XSL-Technologie gut geeignet. [1]

3.1.2 XSLT

XSLT ist eine Transformationssprache zur Übersetzung einer XML-Datei in ein anderes Textformat. Damit können die Referenzen auf Layouts (auch Stylesheets genannt) in die zu formatierten XML-Dokumente eingebunden werden. XSLT bildet zusammen mit

XSL-FO (XSL Formatting Objects) und XPath (XML Path Language) die Extensible Stylesheet Language (XSL). XPath wurde entwickelt, um auf Teile von XML-Dateien zu verweisen. Damit kann man die Baumbestandteile adressieren. [1] [7]

3.1.3 XSL-FO

XSL-FO ist eine Seitenbeschreibungssprache für die Darstellung von XML-Dateien. Damit kann man ein Dokument als Baum mit Formatierungsanweisungen und Stilangaben beschreiben. [1] [6]

Weiterhin berücksichtigt die XSL-FO Erweiterung die Bidirektionalität. Dabei wird das „fo:bidi:override“ Element verwendet, um das Bidirektionalitätsproblem zu lösen. Es wurde im XSL-FO implementiert, um bidirektionale Texte darstellen zu können. [1]

3.1.4 Vorgehensweise

XSL wurde entworfen, um mehrsprachige Dokumente bei der Formatierung zu unterstützen. Dabei wird eine Menge von Layoutobjekten definiert, die die Eigenschaften von den geschriebenen mehrsprachigen Texten beschreiben und darstellen. Dazu braucht man noch einen XSLT-Prozessor, um das XML-Dokument zu formatieren

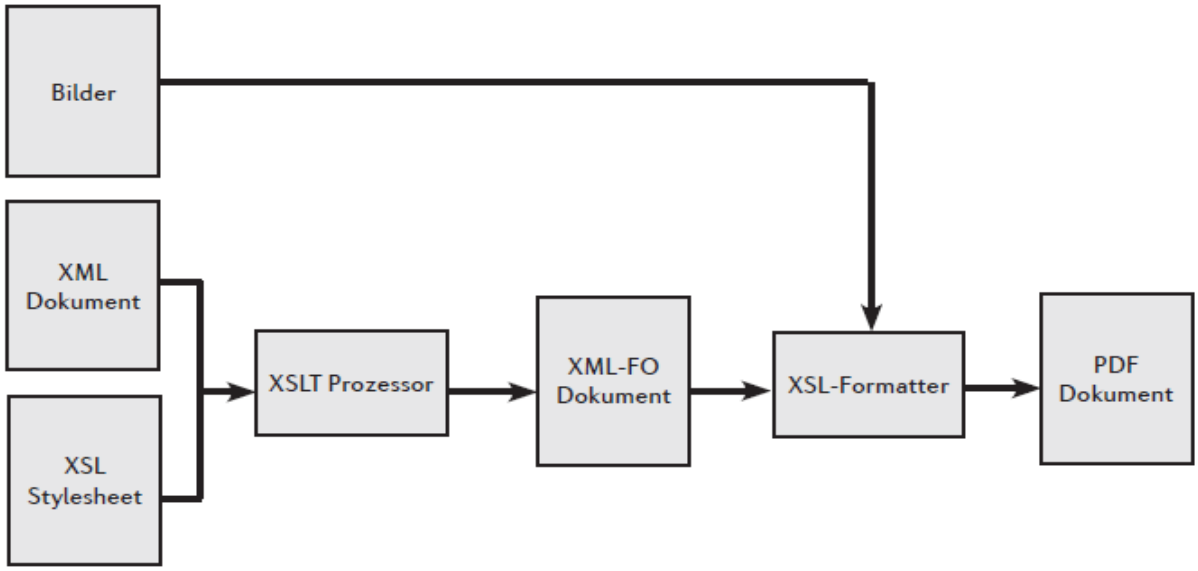


Abbildung 12: Vorgehensweise bei der PDF-Erstellung[6]

In der Regel hat man ein XML Dokument und ein XSL Stylesheet, das ein XSLT Prozes-

sor automatisch in ein XML-FO Datei transformiert. Danach verwendet der XSL-Formatierer diese FO-Datei und die dazugehörigen Grafiken, um z.B. ein PDF zu generieren. Je nach Formatierer kann man sich auch ein SVG oder Postscrip Dokument ausgeben lassen. In dieser Arbeit werden verschiedene Formatierer behandelt [6].

3.2 Werkzeuge

3.2.1 XSLT-Prozessor

Ein XSLT-Prozessor ist eine Software zur Formatierung von XML-Dokumenten mit Hilfe eines XSL-Stytesheets. Dabei werden die Template-Regeln innerhalb des Stylesheets auf das Eingabedokument angewendet und daraus ein neues Dokument erzeugt. Der XSLT Prozessor benötigt zwei Eingabedokumente: das XML Quelldokument und das Stylesheet Dokument. Er liest sie beide und parst sie. Damit der XSLT-Prozessor nicht direkt auf diesen Originaldokumenten arbeitet, generiert das XML Parser zwei Baumrepräsentationen .Diese Baumrepräsentationen werden in einem ersten Schritt erzeugt, dann prüft der XSLT Prozessor, welche Template-Regel des Stylesheets angewendet wird. Als Ausgabe erzeugt er eine Baumrepräsentation vom Ergebnisbaum. Dieser Baum wird dann später serialisiert und kann dann als ein beliebiges Textdokument (XML, HTML, RTF, XSL-FO Designstruktur, etc.) ausgegeben werden. [7]

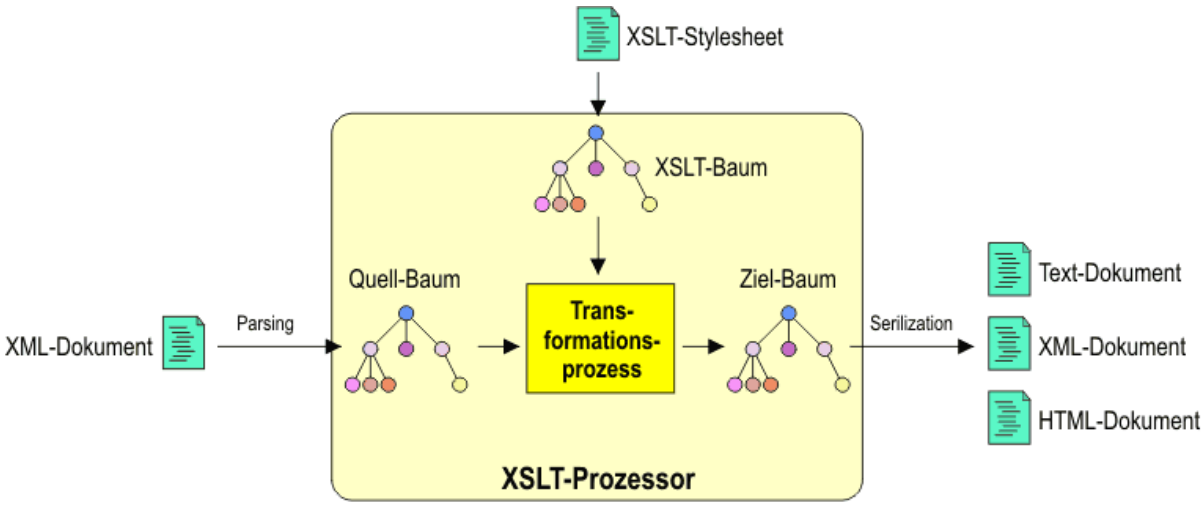


Abbildung 13: XML Verarbeitung mittels eines XSLT Prozessors[33]

3.2.2 XSL-Formatierer

Ein typischer XSL-Formatierer verwendet XSTL-Stylesheets, in denen mit XPath-Ausdrücken auf bestimmte Teile einer XML-Datei verwiesen wird, um diese in XSL-FO zu übersetzen. Das XSLT-Stylesheet wählt aus dem Ursprungsdokument die erforderlichen Elemente aus und transformiert sie in eine Kombination aus XSL-FO-Elementen und –Attributen (XSL-FO-Datei). Diese beschreibt das Erscheinungsbild des Zielformats (z.B. Seitengestaltung, Gestaltung von Spalten, Listen, Tabellen, usw.). Dieser Vorgang wird wiederholt, bis das Ursprungsdokument vollständig abgearbeitet ist. Dann erzeugt ein geeigneter Formatierer (auch XSL-FO-Prozessor genannt) die gewünschte Zieldatei (z.B. PDF, RTF oder PostScript). [1] [8]

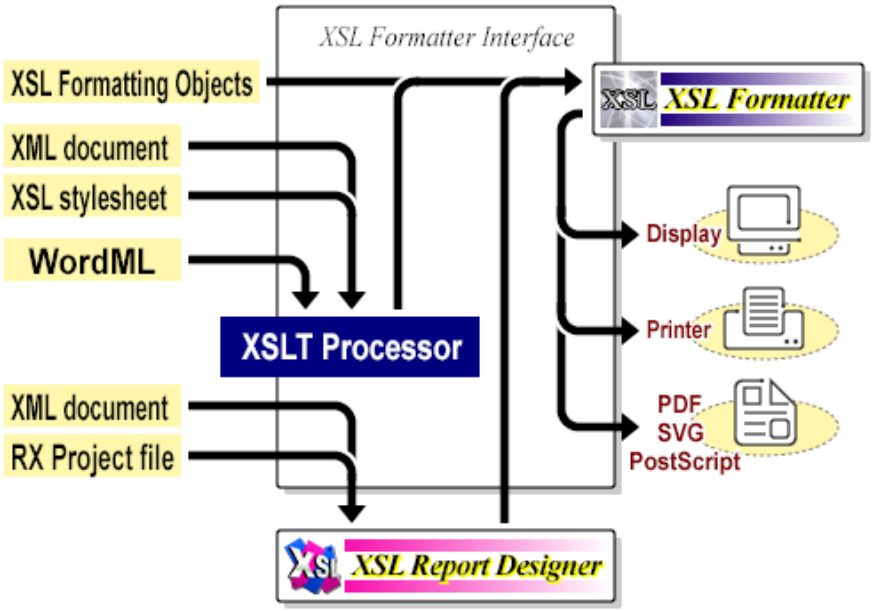


Abbildung 14: XSL Formatierer[8]

3.2.2.1 OpenSource

3.2.2.1.1 FOP

Was ist FOP?

FOP ist die Abkürzung für Formating Objects Processor und ist die bekannteste Open Source Implementation von der standardisierten Sprache XSL-FO. FOP stammt aus dem

Apache XML Projekt und wurde in Java entwickelt. Deshalb ist es lauffähig auf allen wichtigen Plattformen. Der XSL-FO-Prozessor Apache FOP ermöglicht die automatische Generierung qualitativ hochwertiger Druckvorlagen aus XML-Dokumenten und –Daten. [9][10]

Vorteile von FOP:

- Plattformunabhängig
- Die Verwendung eines W3C Standards
- Schnelle Iterationen da nur Templates angepasst werden müssen
- Die Generierung von verschiedenen Dokumenttypen und Inhalten lässt sich an einer Stelle bearbeiten.
- Die Erweiterung um zusätzliche Formate ist möglich

Nachteile von FOP:

- Keine Unterstützung des gesamten Umfangs von XSL:FO 1.0, geschweige denn XSL:FO 1.1
- Keine Beherrschung der Trennung von Wörtern, deswegen sind alle Zeilen auf die angegebene Breite zurecht gezogen
- Variable Tabellen werden nicht unterstützt
- Keine Unterstützung des Einbindens von EPS- bzw. PDF-Grafiken in PDF-Dateien, so dass in diesem Fall auf die Bitmap-Variante zurückgegriffen werden muss [10][11][12][16]

3.2.2.1.2 Xmlroff

Was ist xmlroff?

Xmlroff von Sun Microsystems ist ein schneller und multisprachiger XSL Formatierer. Damit kann man XML- oder XSL-FO-Dateien in PDF oder in Postskriptum umwandeln.

Vorteile von xmlroff:

- Schnell
- Multiplattform
- Mehrsprachig
- Unterstützung der DocBook Formatierung
- Einfache Integration mit anderen Programmen und mit Scripting-Sprachen

Nachteile von xmlroff:

- Keine Unterstützung des gesamten Umfangs von XSL:FO 1.0, geschweige denn XSL:FO 1.1.
- Keine Beherrschung der Trennung von Wörtern, deswegen sind alle Zeilen auf die angegebene Breite zurecht gezogen
- Implementation in C
- Keine Unterstützung des Einbindens von EPS- bzw. PDF-Grafiken in PDF-Dateien, so dass in diesem Fall auf die Bitmap-Variante zurückgegriffen werden muss [11][12]

3.2.2.2 Kommerziell

3.2.2.2.1 AHF

Was ist AHF?

Der AHF Formatierer von Antenna House ist eine professionelle Software. Sie ist führend in der Umsetzung der W3C-Empfehlung für XSL-FO. Sie ist das erste und einzige Werkzeug, das auf der Basis des Standards XSL-FO auch sehr komplexe Aufgaben automatisiert und verarbeitet. Damit kann man XML Daten mittels fortgeschrittenen Funktionen und XSL-FO Stylesheets zu PDF transformieren. [13][14]

Antenna House bietet in ihrer Webseite die Möglichkeit, eine Evaluationslizenz der Produkte und Versionen herunterzuladen. Die Evaluationslizenz ist 90 Tage gültig

und nur für Testzwecke zu verwenden.
Darüber hinaus bietet Antenna House für Endbenutzer sowohl die Standalone als auch die Serverlizenz an. Die Standalone-Lizenz ist für die Installation des Formatters auf einem einzigen Computer beschränkt. Um den Antenna House Formatter auf einem Server verwenden zu können, braucht man eine Serverlizenz. Damit kann eine unbegrenzte Benutzeranzahl den XSL Formatter auf dem Server verwenden.
Für Bildungseinrichtungen bietet Antenna House einen Rabatt von ca. 50% auf die Lizenzpreise. [35]
Der Netto-Preis pro XSL Formatierer V5 Standalone-Lizenz für Linux liegt bei ca. 1.100 Euro; incl. Rabatt bei ca. 650 Euro. Der Netto-Preis pro XSL Formatierer V5 Server- Lizenz für Linux liegt bei ca. 4.100 Euro; incl. Rabatt bei ca. 2400 Euro. [34]

Vorteile von AHF:

- Umfassende Implementation der XSL-FO-Recommendation des W3C
- Die Unterstützung vieler Sprachen (darunter Arabisch, Chinesisch, Japanisch, Hebräisch, Thailändisch)
- Die Unterstützung des Vektorgrafikstandards SVG.
- Der AH Formatierer verfügt über eine grafische Benutzeroberfläche, die sich jedoch auch über diverse Schnittstellen als "Black box" in Prozesse einbinden lässt.
- Der AH Formatierer ist für Windows, Unix und Linux verfügbar und bietet Schnittstellen für Java, .NET, COM und die Kommandozeile.
- Unterstützung der Schnittstellen zu .NET, COM, JAVA, und C/C++. Dadurch lässt sich dieses Produkt gut integrieren und ein vollautomatischer Ablauf erzeugen.

- Direkte Unterstützung aller Windows-Drucker
- Variable Tabellen werden unterstützt
- Die Version 5 des AH Formatierers bietet erstmalig als Alternative neben der Formatierung über XSLT und XSL-FO die Option, CSS als Stylesheet-Sprache zu verwenden. Somit kommt es zu drei alternativen Produkten:
  - Der altbekannte AH XSL Formatierer (formatiert über XSL-FO)
  - Der AH CSS Formatierer (formatiert über CSS)
  - Der AH Formatierer (bietet beide Optionen).
- Auf die Server-Version kann von beliebig vielen Clients zugegriffen werden

Nachteile vom AH Formatierer:

- Der AH Formatierer ist für den Anwender unter Umständen teuer und kompliziert.
- Die AH Formatierer-Preise sind hoch, deswegen wird dieses Tool in Unternehmen nur dann eingesetzt, wenn ein hoher Bedarf an PDFs besteht.
- Keine Beherrschung der Trennung für Wörter, deswegen sind alle Zeilen auf die angegebene Breite zurecht gezogen [13][14][16]

3.2.2.2.2 XEP

Was ist XEP?

XEP ist ein kommerzielles Produkt der Firma RenderX. Es läuft unter Linux, Windows XP, NT, ME, 98 und 2000, die Dateigröße beträgt 3,13 MByte. XEP-Rendering-Engine konvertiert XSL/FO-Dateien gemäß der W3C-Empfehlung in PDF oder Postscript. Die Anwendung wurde komplett in Java implementiert und ist auf jedem System lauffähig, das Java 2 (JDK/JRE Version 1.2.1) unterstützt. Auf der Homepage findet man dort natürlich ausführliche Informationen über die angebotenen Lizenzierungsmodelle.

Zudem kann man verschiedene Versionen für Entwickler, Server und auch eine kostenlose Version zum Testen oder für akademische Zwecke herunterladen. Die Testversion fügt allerdings einen Stempel auf jede Seite oder in der akademischen Version eine Anmerkung in die Notizen des Dokumentes ein. [15][16]

Vorteile von XEP:

- Schnell
- Plattformunabhängig
- Auf die Server-Version kann von beliebig vielen Clients zugegriffen werden
- Mehrsprachig
- Variable Tabellen werden unterstützt
- Fortgeschrittene Funktionen wie z.B. Interaktive Links, Lesezeichen und elektronische Sicherheit.
- Unterstützung der von W3C standardisierten XSL-Version 1.0.
- pro Server Lizenz kosten ca. 4.000 US-$, was für Unternehmen günstig ist, in denen PDFs eine große Verwendung finden. Eine Einzelplatz Lizenz ist für ca. 300 US-$ zu kaufen.

Nachteile von XEP:

- Die XSL-Version 1.1 ist nicht komplett unterstützt worden.
- Keine Beherrschung der Trennung von Wörtern, deswegen sind alle Zeilen auf die angegebene Breite zurecht gezogen. [16]
- Bei der Integration dieses Formatierers in das Projekt geht der Vorteil von „Data-Style-Logik“ Architektur verloren, da man an die Funktionslogik von diesem kommerziellen Produkt gebunden ist. Für unser Projekt ist er auch nicht geeignet.

4 Erstellung von PDF-Dokumenten

4.1 Erstellen von PDF Dokumenten mit Hilfe von FOP

4.1.1 Installation von FOP

Für die Installation von FOP benötigt man folgende Software:

- Eine Java Runtime. Minimal wird Java 1.2.x gebraucht.
- Die benötigten FOP Bibliotheken: FOP-Distribution beinhaltet alle Bibliotheken, die man braucht. Diese liegen in dem „fop/lib“ Verzeichnis. Die Bibliotheken beinhalten:
    - Xerces: ist ein Apache-Projekt zum Parsern und Generieren von XML-Daten. Es stellt einen validierenden Parser für Java bereit. Dadurch werden die W3C-Standards unterstützt. Xerces überprüft, ob ein Dokument wohlgeformt ist und dem DTD entspricht
    - Xalan ist ein XSLT Prozessor der Apache Software Foundation. Xalan ist als Java Version erhältlich. Damit kann man XML und XSL -Dateien in HTML, PDF usw. konvertieren.

Grundlegende FOP Installation

Zuerst sollte man das Quellcodepaket FOP herunterladen, dann die Datei „fop-0.x.x.tar.gz“ in ein Verzeichnis entpacken. Allerdings muss die JVM auf dem Rechner installiert sein, da der Quellcode in Java implementiert wurde. Weiterhin kann man ein kleines Bourne Shell-Skript (z.B. „start_fop.sh“) schreiben, welches Apache FOP startet. Somit ist die Installation von FOP abgeschlossen.

4.1.2 Konfiguration von FOP

Die FOP-Konfiguration kann man in dem folgenden Shell-Skript start_fop.sh schreiben.

```
# Start Xalan
rm -f $HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myFO-Test.fo
export CLASSPATH=$CLASSPATH:/usr/share/java/Xalan-j2.jar:/usr/share/java/xerces-j2.jar:/usr/share/java/Xalan-j2-serializer.jar
java org.apache.Xalan.xslt.Process -in $HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myXML-Test.xml -xsl $HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myXSL-Test.xsl -out $HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myFO-Test.fo
```

```
# set Font Configuration
CFG=$HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/userconfig.xml
if [ ! -f $CFG ]
  then
  echo "<?xml version=\"1.0\"?>" > $CFG
  echo "<fop version=\"1.0\">" >> $CFG
  echo "<renderers>" >> $CFG
  echo "<renderer mime=\"application/pdf\">" >> $CFG
  echo "<filterList>" >> $CFG
  echo "<value>flate</value>" >> $CFG
  echo "</filterList>" >> $CFG
  echo "<fonts>" >> $CFG
  echo "<font metrics-url=\"file://$HOME/workspace/Diplomarbeit_Abde/fonts/arialun.xml\" kerning=\"yes\" embed-url=\"file://$HOME/workspace/Diplomarbeit_Abde/fonts/arialun.ttf\">" >> $CFG
  echo "<font-triplet name=\"ArialUnicodeMS\" style=\"normal\" weight=\"normal\"/>" >> $CFG
  echo "<font-triplet name=\"ArialUnicodeMS\" style=\"normal\" weight=\"400\"/>" >> $CFG
  echo "</font>" >> $CFG
  echo "</fonts>" >> $CFG
  echo "</renderer>" >> $CFG
  echo "</renderers>" >> $CFG
  echo "</fop>" >> $CFG
fi
```

```
#Start FOP

rm -f $HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myPDF-Test.pdf
fop -c $CFG -fo $HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myFO-Test.fo -pdf
$HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myPDF-Test.pdf
```

Zuerst wird die Datei „myFO-Test.fo“ durch den XSLT Prozessor Xalan erzeugt.

```
java org.apache.Xalan.xslt.Process -in $HOME/workspace/Diplomarbeit_Abde/MyFOP-
Test/myXML-Test.xml -xsl $HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myXSL-Test.xsl -
out $HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myFO-Test.fo
```

Dann wird die Font-Konfiguration in der Datei „userconfig.xml“ geschrieben. Die Konfigurationsdatei „userconfig.xml“ liegt unter {fop-Verzeichniss}/conf/. Jedoch kann man es nach seinem Bedarf ändern. Für das Testbeispiel wurde die folgende Konfigurationsdatei verwendet. Dabei kann man diverse Einstellungen vornehmen. Diese Konfigurationseinstellungen regeln das Verhalten des FOP und helfen die entsprechenden Ressourcen zu finden. Um die Mehrsprachigkeit zu gewährleisten, sollen die benutzerdefinierten Unicode-Fonts in FOP eingebunden werden. Dafür sind Fontmetrikinformationen erforderlich. FOP hat einen TTFReader, der die True-Type-Font ausliest und eine geeignete Metrik erzeugt. Diese Metrik wird in der Datei arialun.xml gespeichert. Sie beinhaltet alle Eigenschaften des Bildzeichens.
Nachdem die Metriken für alle gewünschte Fonts erstellt wurden, müssen die Pfade zu diesen Dateien in der Konfigurationsdatei „userconfig.xml“ festgelegt werden.

```
url=\"file://$HOME/workspace/Diplomarbeit_Abde/fonts/arialun.xml\" kerning=\"yes\" embed-
url=\"file://$HOME/workspace/Diplomarbeit_Abde/fonts/arialun.ttf\">" >> $CFG
```

Das Shellskript „start_fop.sh“ erzeugt dann folgende Konfigurationsdatei „userconfig.xml“ [36] [37]:

```
<?xml version="1.0" ?>
<fop version="1.0">
<renderers>
<renderer mime="application/pdf">
        <filterList>
        <value>flate</value>
        </filterList>
        <fonts>
        <font metrics-
        url="file:///home/dptadmin/workspace/Diplomarbeit_Abde/fonts/arialun.xml" kern-
        ing="yes" embed-
        url="file:///home/dptadmin/workspace/Diplomarbeit_Abde/fonts/arialun.ttf">
        <font-triplet name="ArialUnicodeMS" style="normal" weight="normal" />
        <font-triplet name="ArialUnicodeMS" style="normal" weight="400" />
        </font>
        </fonts>
    </renderer>
    </renderers>
     </fop>
```

Jedes Element „<font>“ der Datei .XML beinhaltet die Pfade zu der Fontsatzdatei und die passende Metrik.

```
<font metrics-
url="file:///home/dptadmin/workspace/Diplomarbeit_Abde/fonts/arialun.xml" kern-
ing="yes" embed-
url="file:///home/dptadmin/workspace/Diplomarbeit_Abde/fonts/arialun.ttf">
```

Außerdem enthält das Element „<font>“ ein oder mehrere Tags „<font-triplet>“, damit wird die Menge von Synonymen für diesen Font beschrieben. In diesem Testbeispiel wird der Font „ArialUnicodeMs“ vergeben. Diese Namen werden dann in der XSL Datei für die „font-family“ eingegeben. Für jeden Schriftstil des Schriftsatzes muss ein Element „<font-triplet>“ erstellt werden. In diesem Testbeispiel werden zwei Schriftstile verwendet.

```
<font-triplet name="ArialUnicodeMS" style="normal" weight="normal" />
 <font-triplet name="ArialUnicodeMS" style="normal" weight="400" />
```

Dabei bestimmen die Attribute „style“ und „weight“ den Schriftstile des Schriftsatzes.
Somit ist die Konfiguration von FOP abgeschlossen. Der Formatierer FOP wird ausgeführt.

```
#Start FOP
rm -f $HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myPDF-Test.pdf
fop -c $CFG -fo $HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myFO-Test.fo -pdf
$HOME/workspace/Diplomarbeit_Abde/MyFOP-Test/myPDF-Test.pdf
```

Dabei werden die Dateien „userconfig.xml“ und „MyFO-Test.fo“ eingebunden. Als Ausgabe wird die Testbeispieldatei „myPDF-Test.pdf“ erstellt.

4.1.3 Testbeispiel mit FOP

Das PDF-Dokument wurde stabil erstellt (siehe die Abbildung 15).

Abbildung 15: FOP-Testbeispiel [35]

Dabei wurden verschiedene Schreibmöglichkeiten getestet. z. B lateinischer Text mit arabischen Sprachzeichen. Das Schreiben von rechts nach links stellt ein Problem dar. Dabei wurden die Absatzausrichtung und Textrichtung von rechts nach links nicht

unterstützt. FOP ist nicht in der Lage, bidirektionalen Text darzustellen, die Verbindung der einzelnen Buchstaben funktioniert somit nicht. Die bidirektionalen Texte erscheinen daher ebenfalls als einzelne unverbundene Buchstaben von links nach rechts geschrieben, was natürlich falsch ist.

4.1.4 Fazit

Das Testergebnis war total negativ, weil es unseren Anforderungen nicht entsprach. FOP bietet leider keine Lösungsmöglichkeiten für die Problematiken der Darstellung arabischer Schrift. Damit ist es für unsere Projekte überhaupt nicht geeignet.

4.2 Erstellen von PDF Dokumenten mit Hilfe von XRF

4.2.1 Installation von XRF

Für die Installation von XRF benötigt man folgende Software:

- Libgnomeprint
- Pango
- Cairo
- libxml2
- libxslt
- GTK2
- GLib
- GObject

Es wird auch empfohlen folgende Abhängigkeiten zu installieren:

- libxslt1-dev
- libpango1.0-dev
- libglib2.0-dev
- libcairo2-dev
- libgtk2.0-dev

Grundlegende XRF Installation

Zuerst muss man eine Lizenz erwerben. Damit kann man das Quellcodepaket z.B. „**xmlroff-0.5.3.tar.gz**" problemlos herunterladen. Danach muss man es in einem Verzeichnis so entpacken: **tar zxvf xmlroff-0.6.2.tar.gz**. Dann sollte der Befehl „**./configure**" „**make**" und „**make install**" ausgeführt werden, um das Paket zu installieren. Weiterhin kann man ein kleines Bourne Shell-Skript z.B. „**start_xrf.sh**" schreiben, welches XRF startet. Somit ist die Installation von XRF abgeschlossen.

4.2.2 Konfiguration von XRF

Die XRF-Konfiguration kann man im folgenden Shell-Skript „start_xrf.sh" beschreiben.

```
# Start Xalan

rm -f $HOME/workspace/Diplomarbeit_Abde/src/MyXRF-Test/myFO-Test.fo
export CLASSPATH=$CLASSPATH:/usr/share/java/Xalan-j2.jar:/usr/share/java/xerces-
j2.jar:/usr/share/java/Xalan-j2-serializer.jar
java org.apache.Xalan.xslt.Process -in $HOME/workspace/Diplomarbeit_Abde/src/MyXRF-
Test/myXML-Test.xml -xsl $HOME/workspace/Diplomarbeit_Abde/src/MyXRF-Test/myXSL-
Test.xsl -out $HOME/workspace/Diplomarbeit_Abde/src/MyXRF-Test/myFO-Test.fo

#Start XRF

rm -f $HOME/workspace/Diplomarbeit_Abde/src/MyXRF-Test/myPDF-Test.pdf
/home/akarmoun/workspace/Diplomarbeit_Abde/docs/xmlroff-0.6.2/xmlroff/xmlroff myFO-
Test.fo -o myPDF-Test.pdf
```

Zuerst wird die Datei „myFO-Test.fo" durch den XSLT Prozessor Xalan erzeugt. Danach wird die Datei „xmlroff" ausgeführt und die Datei „MyFO-Test.fo" eingebunden. Als Ausgabe wird das Testbeispieldatei „myPDF-Test.pdf" erstellt.

4.2.3 Testbeispiel mit XRF

Das PDF-Dokument wurde fehlerfrei und stabil erstellt (siehe Abbildung 16).

Abbildung 16: XRF-Testbeispiel [35]

Mit dem Testbeispiel wurden verschiedene Schreibmöglichkeiten getestet. XRF unterstützt bidirektionalen Text. Deshalb funktionieren die Verbindungen der einzelnen Buchstaben einwandfrei. Allerdings war die "links-rechts" Schreibrichtung teilweise fehlerhaft, und es gab auch ein Problem bei der Darstellung der Ligatur und ḥarakāt.

4.2.4 Fazit

Das Testergebnis war positiv, weil es manche Anforderungen einlöste. Leider unterstützt XRF nicht alle Lösungen für die Problematiken der Darstellung arabischer Schrift. Damit ist es für unsere Projekte nicht gut geeignet.

4.3 Erstellen von PDF Dokumenten mit Hilfe von AHF

4.3.1 Installation von AHF

Für die Installation von AHF benötigt man folgende Software:

- Eine Java Runtime. Minimal wird Java 1.2.x gebraucht
- Die benötigten AHF-Bibliotheken: AHF-Distribution beinhaltet alle Bibliotheken, die man braucht. Diese liegen in dem „AHFormatierer/lib“ Verzeichnis. Die Bibliotheken beinhalten: Xerces und Xalan.

Grundlegende AHF Installation

Zuerst sollte man das Quellcodepaket „AHFormatiererV51-5.1E-M1.i386.rpm.gz” herunterladen und es in einem Verzeichnis entpacken. Allerdings muss die JVM auf dem Rechner installiert sein. Dann sollte der Befehl „ rpm -i [--nodeps] [The rpm file name] [--prefix deistination-path-to-install]“ ausgeführt werden, um das Paket zu installieren. Weiterhin kann man ein kleines Bourne Shell-Skript (z.B. „start_ahf.sh“) schreiben, welches AHF startet. Somit ist die Installation von AHF abgeschlossen.

4.3.2 Konfiguration von AHF

Die AHF-Konfiguration kann man im folgenden Shell-Skript start_ahf.sh beschreiben.

```
# Start Xalan
rm -f $HOME/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myFO-Test.fo
export CLASSPATH=$CLASSPATH:/usr/share/java/Xalan-j2.jar:/usr/share/java/xerces-
j2.jar:/usr/share/java/Xalan-j2-serializer.jar
java org.apache.Xalan.xslt.Process -in $HOME/workspace/Diplomarbeit_Abde/src/MyAHF-
Test/myXML-Test.xml -xsl $HOME/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myXSL-
Test.xsl -out $HOME/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myFO-Test.fo
```

```
#Start AHF

rm -f $HOME/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myPDF-Test.pdf

/usr/local/src/AHFormatiererV51_64/run.sh -d
/home/akarmoun/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myFO-Test.fo -o
/home/akarmoun/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myPDF-Test.pdf
```

Zuerst wird die Datei „myFO-Test.fo“ durch den XSLT Prozessor Xalan erzeugt.

```
# Start Xalan

rm -f $HOME/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myFO-Test.fo
export CLASSPATH=$CLASSPATH:/usr/share/java/Xalan-j2.jar:/usr/share/java/xerces-
j2.jar:/usr/share/java/Xalan-j2-serializer.jar
java org.apache.Xalan.xslt.Process -in $HOME/workspace/Diplomarbeit_Abde/src/MyAHF-
Test/myXML-Test.xml -xsl $HOME/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myXSL-
Test.xsl -out $HOME/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myFO-Test.fo
```

Dann wird die Font-Konfiguration in der Datei „Font-Config.xml“ geschrieben und in dem Verzeichnis „AHF51_64_HOME/etc“ gespeichert. Jedoch kann man es nach seinem Bedarf ändern. Für das Testbeispiel wurde die folgende Konfigurationsdatei verwendet. [36] [37]

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!-- DOCTYPE font-config SYSTEM "font-config.dtd" -->
  <font-config>
  <font-folder path="/usr/local/src/AHFormatiererV51_64/fonts">
    <font-alias file="arialuni.ttf">
        <alias family-name="ArialUnicodeMS" />
  </font-alias>
 </font-folder>
 </font-config>
```

Dabei kann man diverse Einstellungen vornehmen. Diese Konfigurationseinstellungen regeln das Verhalten des AHF und helfen die entsprechenden Ressourcen zu finden. Sie wird in dem Startshellskript „run.sh“ aufgerufen.

```
AHF51_64_FONT_CONFIGFILE=${AHF51_64_ETC_FOLDER}/font-config.xml
export AHF51_64_FONT_CONFIGFILE
```

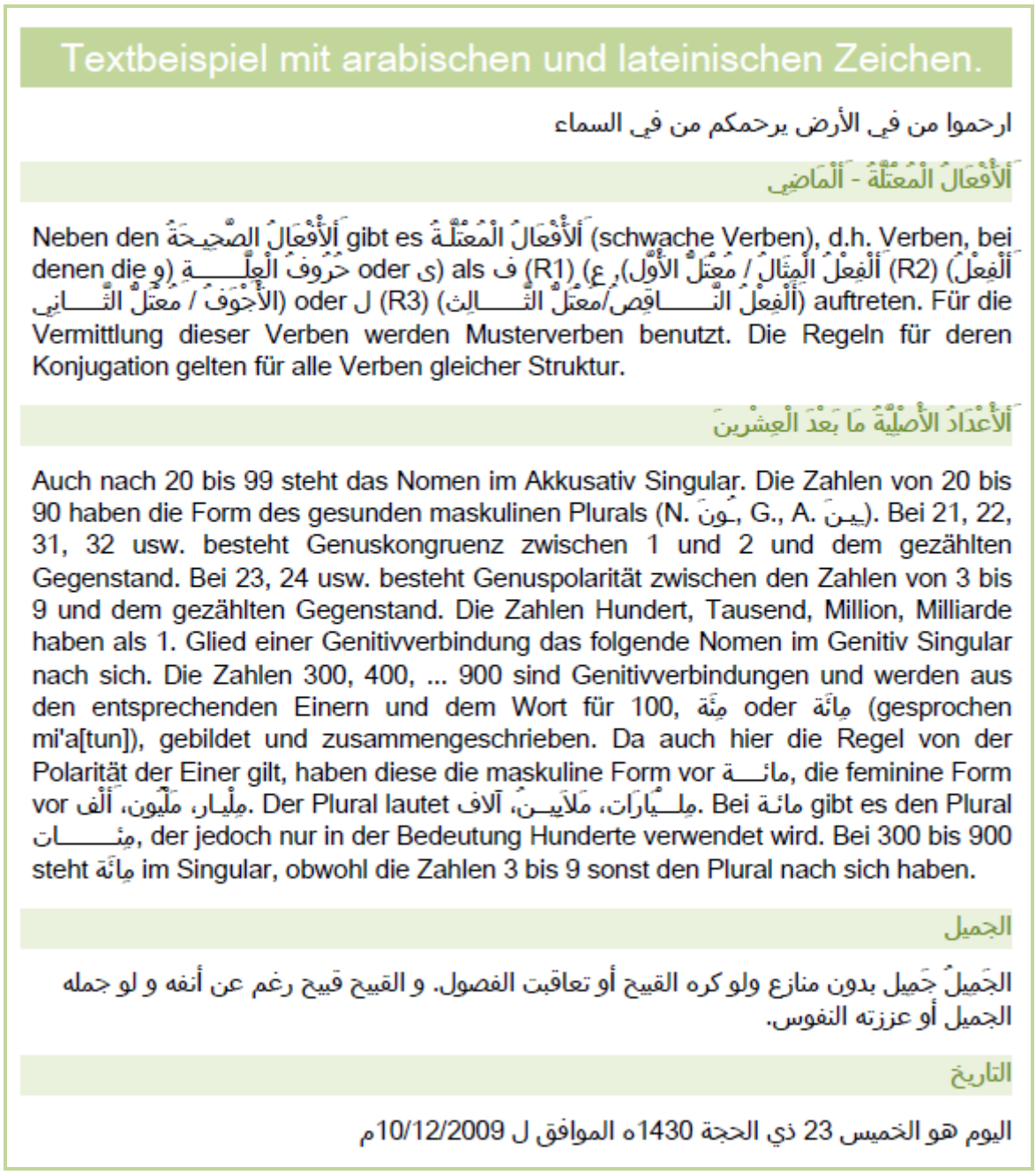
Um die Mehrsprachigkeit zu gewährleisten, werden die benutzerdefinierten Unicode-Fonts eingebunden. Dafür müssen die Pfade zu diesen Dateien in der Konfigurationsdatei „Font-Config.xml“ festgelegt werden. Das folgende Element: <font-folder path> beinhaltet die Pfade zu der Fontsatzdatei. Weiterhin wird in diesem Testbeispiel der folgende Font „ArialUnicodeMs“ verwendet. <alias family-name="ArialUnicodeMS" />: Dieser Name wird dann in die XSL-Datei für die „font-family“ eingegeben. Somit ist die Konfiguration von AHF abgeschlossen und es wird mit dem folgenden Befehl ausgeführt.

```
#Start AHF
rm -f $HOME/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myPDF-Test.pdf
/usr/local/src/AHFormatiererV51_64/run.sh -d
/home/akarmoun/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myFO-Test.fo -o
/home/akarmoun/workspace/Diplomarbeit_Abde/src/MyAHF-Test/myPDF-Test.pdf
```

Dabei wird die Datei „run.sh“ ausgeführt und die Datei „MyFO-Test.fo“ eingebunden. Als Ausgabe wird die Testbeispieldatei „myPDF-Test.pdf“ erstellt.

4.3.3 Testbeispiel mit AHF

Das PDF-Dokument wurde fehlerfrei und stabil erstellt (siehe Abbildung 17).



Textbeispiel mit arabischen und lateinischen Zeichen.

ارحموا من في الأرض يرحمكم من في السماء

اَلْأَفْعَالُ الْمُعْتَلَّةُ - اَلْمَاضِي

Neben den اَلْأَفْعَالُ الصَّحِيحَةُ gibt es اَلْأَفْعَالُ الْمُعْتَلَّةُ (schwache Verben), d.h. Verben, bei denen die حُرُوفُ الْعِلَّةِ (و oder ى) als ف (R1) ع, (اَلْفِعْلُ الْمِثَالُ / مُعْتَلُّ الْأَوَّلِ) (R2) اَلْفِعْلُ (اَلْفِعْلُ النَّاقِصُ/مُعْتَلُّ الثَّالِثِ) (R3) ل oder (الْأَجْوَفُ / مُعْتَلُّ الثَّانِي) auftreten. Für die Vermittlung dieser Verben werden Musterverben benutzt. Die Regeln für deren Konjugation gelten für alle Verben gleicher Struktur.

اَلْأَعْدَادُ الْأَصْلِيَّةُ مَا بَعْدَ الْعِشْرِينَ

Auch nach 20 bis 99 steht das Nomen im Akkusativ Singular. Die Zahlen von 20 bis 90 haben die Form des gesunden maskulinen Plurals (N. ـُونَ, G., A. ـِينَ). Bei 21, 22, 31, 32 usw. besteht Genuskongruenz zwischen 1 und 2 und dem gezählten Gegenstand. Bei 23, 24 usw. besteht Genuspolarität zwischen den Zahlen von 3 bis 9 und dem gezählten Gegenstand. Die Zahlen Hundert, Tausend, Million, Milliarde haben als 1. Glied einer Genitivverbindung das folgende Nomen im Genitiv Singular nach sich. Die Zahlen 300, 400, ... 900 sind Genitivverbindungen und werden aus den entsprechenden Einern und dem Wort für 100, مِئَة oder مِائَة (gesprochen mi'a[tun]), gebildet und zusammengeschrieben. Da auch hier die Regel von der Polarität der Einer gilt, haben diese die maskuline Form vor مائـــة, die feminine Form vor مِلْيَار، مَلْيُون، أَلْف. Der Plural lautet آلاف، مَلَايِـين، مِلْـيَارَات. Bei مائة gibt es den Plural مِئـــات, der jedoch nur in der Bedeutung Hunderte verwendet wird. Bei 300 bis 900 steht مِائَة im Singular, obwohl die Zahlen 3 bis 9 sonst den Plural nach sich haben.

الجميل

الجَمِيلُ جَمِيل بدون منازع ولو كره القبيح أو تعاقبت الفصول. و القبيح قبيح رغم عن أنفه و لو جمله الجميل أو عززته النفوس.

التاريخ

اليوم هو الخميس 23 ذي الحجة 1430ه الموافق ل 10/12/2009م

Abbildung 17: AHF-Testbeispiel [35]

Dabei wurden verschiedene Schreibmöglichkeiten getestet. z. B lateinischer Text mit arabischen Sprachzeichen. Das Schreiben von rechts nach links funktioniert problemlos. Dabei wurden die Absatzausrichtung und Textrichtung von rechts nach links unter-

stützt. AHF war in der Lage, bidirektionalen Text darzustellen. Folglich haben die Verbindungen der einzelnen Buchstaben gut funktioniert. Auch die so genannten Ligaturen wurden einwandfrei formatiert.

4.3.4 Fazit

Das Testergebnis war positiv, weil es die Anforderungen erfüllte. AHF unterstützt Lösungsmöglichkeiten für alle Problematiken der Darstellung arabischer Schrift. Damit ist es für unsere Projekte gut geeignet.

4.4 Erstellen von PDF Dokumenten mit Hilfe von XEP

4.4.1 Installation von XEP

Für die Installation von XEP benötigt man folgende Software:

- Eine Java Runtime. Minimal wird Java 1.2.x gebraucht
- Die benötigten XEP-Bibliotheken: XEP-Distribution beinhaltet alle Bibliotheken, die man braucht. Diese liegen in dem „XEP_HOME/lib“ Verzeichnis.

Grundlegende XEP Installation

Zuerst muss man eine Lizenz erwerben. Damit kann man das Quellcodepaket z.B. „xep-4.17-20091204-trial.zip” problemlos herunterladen. Danach muss man es in einem Verzeichnis entpacken. Dann sollte der Befehl „ java -jar setup-4.17-20091204-trial.jar“ ausgeführt werden, um das Paket zu installieren. Allerdings muss die JVM auf dem Rechner installiert sein, da der Quellcode in Java implementiert wurde. Weiterhin kann man ein kleines Bourne Shell-Skript z.B. „start_xep.sh“ schreiben, welches XEP startet. Somit ist die Installation von XEP abgeschlossen.

4.4.2 Konfiguration von XEP

Die XEP-Konfiguration kann man im folgenden Shell-Skript „start_xep.sh“ beschreiben.

```
# Start Xalan

rm -f $HOME/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myFO-Test.fo
export CLASSPATH=$CLASSPATH:/usr/share/java/Xalan-j2.jar:/usr/share/java/xerces-
j2.jar:/usr/share/java/Xalan-j2-serializer.jar
java org.apache.Xalan.xslt.Process -in $HOME/workspace/Diplomarbeit_Abde/src/MyXEP-
Test/myXML-Test.xml -xsl $HOME/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myXSL-
Test.xsl -out $HOME/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myFO-Test.fo

#Start XEP

rm -f $HOME/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myPDF-Test.pdf

/home/akarmoun/workspace/Diplomarbeit_Abde/docs/RenderX_XEP/xep
/home/akarmoun/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myFO-Test.fo
/home/akarmoun/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myPDF-Test.pdf
```

Zuerst wird die Datei „myFO-Test.fo“ durch den XSLT Prozessor Xalan erzeugt.

```
# Start Xalan

rm -f $HOME/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myFO-Test.fo
export CLASSPATH=$CLASSPATH:/usr/share/java/Xalan-j2.jar:/usr/share/java/xerces-
j2.jar:/usr/share/java/Xalan-j2-serializer.jar
java org.apache.Xalan.xslt.Process -in $HOME/workspace/Diplomarbeit_Abde/src/MyXEP-
Test/myXML-Test.xml -xsl $HOME/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myXSL-
Test.xsl -out $HOME/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myFO-Test.fo
```

Dann wird die Font-Konfiguration in der Datei „xep.xml“ geschrieben und in dem Verzeichnis „XEP_HOME“ gespeichert. Allerdings kann man es nach seinem Bedarf ändern. Dabei kann man diverse Einstellungen vornehmen. Diese Konfigurationseinstellungen regeln das Verhalten des XEP und helfen die entsprechenden Ressourcen zu finden. Für das Testbeispiel wurde die folgende Konfigurationsdatei verwendet. [36] [37]

```
<font-group path="/usr/local/src/fonts" label="Windows TrueType" embed="true" subset="true">
        <font-family name="arialuni">
           <font>
                    <font-data ttf="arialuni.ttf" />
          </font>
        </font-family>
  <font-alias name="ArialUnicodeMS" value="arialuni" />
</font-group>
```

Sie wird in dem folgenden Startshellskript „xep.sh“ aufgerufen.

```
"$JAVA_HOME/bin/java" \
        -classpath "$CP" \
        "-Dcom.renderx.xep.CONFIG=$XEP_HOME/xep.xml" \
        com.renderx.xep.XSLDriver "$@"
```

Um die Mehrsprachigkeit zu gewährleisten, wird der benutzerdefinierte Unicode-Font eingebunden. Dafür müssen die Pfade zu diesen Dateien in der Konfigurationsdatei „Font-Config.xml“ festgelegt werden. Das Element <font-group path> beinhaltet die Pfade zu der Fontsatzdatei. In diesem Testbeispiel wird der folgende Font „ArialUnicodeMs“ verwendet. <font-alias name="ArialUnicodeMS" value="arialuni" /> Dieser Name wird dann in die XSL-Datei für die „font-family“ eingegeben. Somit ist die Konfiguration von XEP abgeschlossen und es wird mit dem folgenden Befehl ausgeführt.

```
#Start XEP

rm -f $HOME/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myPDF-Test.pdf
/home/akarmoun/workspace/Diplomarbeit_Abde/docs/RenderX_XEP/xep
/home/akarmoun/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myFO-Test.fo
/home/akarmoun/workspace/Diplomarbeit_Abde/src/MyXEP-Test/myPDF-Test.pdf
```

Dabei wird die Datei „xep.sh“ ausgeführt und die Datei „MyFO-Test.fo“ eingebunden. Als Ausgabe wird das Testbeispieldatei „myPDF-Test.pdf“ erstellt.

4.4.3 Testbeispiel mit XEP

Das PDF-Dokument wurde fehlerfrei und stabil erstellt (siehe Abbildung 18).

Abbildung 18: XEP-Testbeispiel [35]

Mit dem Testbeispiel wurden verschiedene Schreibmöglichkeiten getestet. XEP unter-

stützt bidirektionalen Text. Deshalb funktionieren die Verbindungen der einzelnen Buchstaben einwandfrei. Allerdings war die "links-rechts" Schreibrichtung teilweise fehlerhaft, und es gab auch ein Problem bei der Darstellung der Ligatur und ḥarakāt.

4.4.4 Fazit

Das Testergebnis war positiv, weil es manche Anforderungen einlöste. Leider unterstützt XEP nicht alle Lösungen für die Problematiken der Darstellung arabischer Schrift. Damit ist es für unsere Projekte nicht gut geeignet.

5 Resultat

5.1 Evaluierung der Formatierer

5.1.1 Gegenüberstellung der Formatierer bezüglich der Unterstützung der arabischen Darstellungsprobleme

Bedeutung der Kürzel

J = Ja N = Nein P = Partiell

Checkliste der arabischen Darstellungsproblematiken				
Name	Antenna House	FOP	RenderX XEP	Xmlroff
	Supported	Supported	Supported	Supported
Right-to-Left-Schriftrichtung	J	N	J	J
Bidirektionalität	J	N	P	P
Buchstabenverbundenheit	J	N	J	J
ḥarakāt (arabische Vokale)	J	N	P	P
Arabische Datumsangaben	J	N	N	J
Ligatur	J	N	N	N

Tabelle 1 Unterstützung der Darstellung des Arabischen

5.1.2 Gegenüberstellung der Formatierer bezüglich der Unterstützung der technischen Darstellungsanforderungen

Bedeutung der Kürzel

J = Ja N = Nein P = Partiell

Checkliste der technischen Darstellungsanforderungen

Name	Antenna House	FOP	RenderX XEP	Xmlroff
	Supported	Supported	Supported	Supported
Mehrsprachigkeit	J	P	J	J
Einbindung zusätzlicher Fonts	J	J	J	-
Schnittstellen zu Java	J	J	J	N
Plattformunabhängigkeit	J	J	J	J
Integrationsfähigkeit	J	J	J	-
Data-Logic-Style-Architektur	P	J	P	-

Tabelle 2 Unterstützung der technischen Darstellungsanforderungen

5.2 Fazit und Ausblick

Das Ziel der Diplomarbeit war die Untersuchung, Evaluierung und Integration von Formatierern in das Projekt, die eine automatische Generierung von PDF-Dokumenten unter besonderer Berücksichtigung der Ausgabe arabischer und gemischtsprachiger Texte mit unterschiedlichen Schriftausrichtungen ermöglichen.

Zur Erstellen von PDF-Dokumenten wurden Schnittstellen in das Projekt integriert, die den Benutzern ermöglichen, Informationen aus der Webanwendung im PDF-Format anzuzeigen, zu speichern oder auszudrucken.

Im ersten Teil der Diplomarbeit wurden die Besonderheiten der arabischen Sprache, die systematische Beschreibung der Problematiken der Darstellung arabischer Schrift und die technischen Lösungsmöglichkeiten zur Bewältigung solcher Problematiken behandelt. Die Hauptprobleme bei der arabischen Schriftdarstellung waren folgende:

- Problem der Right-to-Left-Schriftrichtung
- Problem der bidirektionalen Texte
- Problem der Buchstabenverbindung
- Problem der arabischen Fonts
- Problem der Ligatur

Im zweiten Teil wurden die Techniken und Werkzeuge zur PDF-Dokumentenerstellung analysiert. Dabei wurden vier wichtige Formatierer behandelt. Die Auswahl basiert auf folgenden Aspekten:

- Unterstützung von Mehrsprachigkeit
- Unterstützung der Einbindung zusätzlicher Fonts
- Unterstützung von Data-Logic-Style-Architektur
- Unterstützung von XSL-FO Standards
- Plattformunabhängigkeit
- Integrationsfähigkeit

Im praktischen Teil der Arbeit wurden die Test-Versionen der kommerziellen Formatierer und die Vollversionen der Open Source Formatierer aus dem Internet heruntergeladen, installiert und konfiguriert und das Ergebnis anhand eines Testbeispiels angezeigt.

Die Ergebnisse der Testbeispiele zeigten, dass der einzigste PDF-Formatierer, der alle Anforderungen unterstützt, AHF von Antenna House war.
Darüber hinaus wurde festgestellt, dass der Funktionsumfang und der Automatisierungsgrad, den man mit dem Produkt AHF von Antenna House erreicht, macht eine Investition in dieses Produkt sinnvoll.
In der Entwicklungs- und Testphase hat das Produkt AHF PDF-Dokumente stabil und zuverlässig erzeugt, so dass es auch erfolgreich im Projekt eingesetzt werden kann.
Auf anderer Seite wurde anhand der Ergebnisse festgestellt, dass es zurzeit nicht empfehlenswert ist, auf Open-Source Produkte zurückzugreifen, besonders für einen Einsatz in großen Projekten, in denen die arabische Schrift, Zuverlässigkeit, Sicherheit und die Verfügbarkeit der Informationen eine große Rolle spielen, weil es bei da immer noch Probleme gibt.
Das Produkt FOP befindet sich z.B. noch in der Entwicklungsphase und unterstützt den XSL-FO-Standard nicht vollständig, besonders das „fo:bidi:override“-Element.
In Folge dessen kam es beim Erstellen des PDF-Dokuments zu falschen Darstellungen von „Right to Left“ Sprachen.
Wie weit das FOP-Team bei der Implementierung dieses Elementes ist, konnten wir nicht wissen, da Kontaktversuche mit der Firma leider nicht erfolgreich waren.
Ein weiterer Aspekt bei der Auswahl ist die Möglichkeit des Zugangs zu den Informationen über das Produkt. Die Webseite von Antenna House ist im Vergleich zur Webseite von RenderX etwas übersichtlicher. Das Suchen von Informationen bei der Webseite von Antenna House ist leichter, schneller und umfangreicher als auf der RenderX Webseite.
Das AHF-Produkt von Antenna House hat auch den weiteren Vorteil, dass es auf jahrelanger Erfahrung basiert.

Literaturverzeichnis

[1] Solution for Multilingual Publishing by Unicode and XSL, Antenna House, Inc, January 2, 2004

[34] Lizenzkosten für Antenna House-Produkte, doctronic GmbH & Co. KG, Mai 2010

[35] Eckehard Schulz, Günther Krahl, Wolfgang Reuschel, Lehrbuch des modernen Arabisch, Langenscheidt Verlag, Januar 2005

[36] Manfred Krüger, XSL-FO - verstehen und anwenden: XML-Verarbeitung für PDF und Druck, Dpunkt Verlag, Juli 2006

[37] Frank Bongers, XSLT 2.0 und XPath 2.0, Galileo Press Verlag, Januar 2008

Internetpräsenz:

[2] Franz Morcinek: Arabische Wurzel die Mathematik der arabischen Sprache (abgerufen am 24.05.10): http://public.beuth-hochschule.de/~morcinek/arab.pdf

[3] Übersetzungsbüro und Kanzlei CANOUN: Die Geschichte der arabischen Sprache (abgerufen am 24.05.10): http://www.canoun.de/resources/Arabische+Sprache-Web.doc

[4] Arabia-Institut: Die Schreibung des arabischen Alphabets (abgerufen am 24.05.10): http://www.arabia-institut.de/documents/arabische-schrift.pdf

[5] Arabische Schrift und Sprache: Arabia-Institut: Die Kunst der Kalligrafie (abgerufen am 24.05.10): http://www.chj.de/Arab-Kali.html

[6] Björn Gottwald, Frank Robnik, Rainer Wagner: XSL Formatting Objects (abgerufen am 24.05.10): http://www.hs-augsburg.de/~rwagner/Seminararbeit_XSL-FO.pdf

[7] Daniel Brügge: XSLT Prozessoren: http://idleaf.com/resources/download/K2XRFLOV8VSI65S7V6RN

[8] Antenna House: XSL Formatierer V4 (abgerufen am 24.05.10): http://www.antennahouse.com/product/axfo40/axfo4top.htm

[9] Jeremias Märki: Automatisierte Dokumentenproduktion mit Apache FOP (abgerufen am 24.05.10): http://www.openexpo.ch/fileadmin/documents/2008Bern/Slides/46.pdf

[10] Michael Knümann 2004: PDF-Generierung mit Java und Apache FOP http://www.knuemann.de/information/pdfgenerierung.pdf

[11] Project TextGrid: Modulare Plattform für verteile und kooperative wissenschaftliche Textdatenverarbeitung- ein Community-Grid für die Geisteswissenschaften (abgerufen am 24.05.10): http://www.textgrid.de/fileadmin/TextGrid/reports/TextGrid-R1_4_Publishing.pdf

[12] Martin Gieseting, Oliver Vornberger: Ein Generator für PDF und HTML (abgerufen am 24.05.10): http://www-lehre.informatik.uni-osnabrueck.de/misc/ctdemo/artikel.txt

[13] Antenna House: Professional Formatting Solutions (abgerufen am 24.05.10): http://www.antennahouse.com/

[14] Date2type: Der XSL Formatierer (Antenna House) (abgerufen am 24.05.10): http://www.data2type.de/software/xsl-Formatierer

[15] PDF "on-the-fly": Von strukturierten Daten zum PDF mit XSL/FO (abgerufen am 24.05.10): http://212.227.4.36/pdf/pp_03_05_d1e.pdf

[16] Tanja Schniederberend: Cross Media Publishing von Lehrmaterialien mit XML Schema & XSL-Transformationen (abgerufen am 24.05.10): www-lehre.informatik.uniosnabrueck.de/~tschnied/Diplomarbeit.pdf

[17] Pilotprojekt zur datenbankgestützten Erschließung und digitalen Bereitstellung der arabischen, persischen und türkischen Handschriften der Universitätsbibliothek (abgerufen am 24.05.10): http://www.islamic-manuscripts.de und http://www.refaiya.uni-leipzig.de

[18] Projekt zur datenbankgestützten Erschließung und digitalen Bereitstellung der orientalischen Handschriften in Indonesien (abgerufen am 24.05.10): http://acehms.dl.uni-leipzig.de , http://manassa.dl.uni-leipzig.de und http://javams.dl.uni-leipzig.de

[19] MyCoRe-System zur Entwicklung von Dokumenten- und Publikationsservern, Archivanwendungen, Sammlungen von Digitalisaten oder vergleichbaren Repositorien (abgerufen am 24.05.10): http://www.mycore.de/

[20] Unterstützende XSL-Fo-Standards von Apache FOP (abgerufen am 24.05.10): http://xmlgraphics.apache.org/fop/compliance.html

[21] Unterstützende XSL-Fo-Standards von RenderX (abgerufen am 24.05.10): http://www.renderx.com/reference.html#XslFoConformance

[22] Arabisches Alphabet (abgerufen am 24.05.10): http://de.wikipedia.org/wiki/Arabisches_Alphabet

[23] Ligatur (abgerufen am 24.05.10): http://de.wikipedia.org/wiki/Ligatur_(Typografie)

[24] ḥarakāt (abgerufen am 24.05.10): http://de.wikipedia.org/wiki/Datei:Harakat_pashto.svg

[25] Kūfī (abgerufen am 24.05.10): http://de.wikipedia.org/wiki/Kufi

[26] Nasḫī (abgerufen am 24.05.10): http://www.schriften-lernen.de/Schrift/Arab/Kalligrafie01.htm

[27] Ṯuluṯ (abgerufen am 24.05.10):
http://de.wikipedia.org/wiki/Thuluth

[28] Req'a (abgerufen am 24.05.10):
http://www.alkhulaki.com/vb/showthread.php?t=1738

[29] Taʿlīq (abgerufen am 24.05.10):
http://www.alkhulaki.com/vb/showthread.php?t=1738

[30] Nastaʿlīq (abgerufen am 24.05.10):
http://de.academic.ru/dic.nsf/dewiki/93346

[31] Dīwānī (abgerufen am 24.05.10):
http://de.wikipedia.org/wiki/Diwani

[32] UTF8 (abgerufen am 25.05.10):
http://de.wikipedia.org/wiki/UTF-8

[33] XSLT-Prozessor (abgerufen am 25.05.10):
http://www.somplatzki.de/xslt/artikel/der_xslt_prozessor.html

Anhang: Gegenüberstellung der Formatierer bezüglich der Unterstützung von XSL-FO Standards

Bedeutung der Kürzel

J = Ja N = Nein P = Partiell

Checkliste der Unterstützung von XSL-FO Objects

6.4 Declarations and Pagination and Layout Formatting Objects

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
6.4.2 fo:root	Basic	J	J	J
6.4.3 fo:declarations	Basic	J	N	N
6.4.4 fo:color-profile	Extended	J	N	N
6.4.5 fo:page-sequence	Basic	J	J	J
6.4.6 fo:page-sequence-wrapper	Basic	J	-	-
6.4.7 fo:layout-master-set	Basic	J	J	J
6.4.8 fo:page-sequence-master	Basic	J	J	J
6.4.9 fo:single-page-master-reference	Basic	J	J	J
6.4.10 fo:repeatable-page-master-reference	Basic	J	J	J
6.4.11fo:repeatable-page-master-alternatives	Extended	J	J	J
6.4.12 fo:conditional-page-master-reference	Extended	J	J	J

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
6.4.13 fo:simple-page-master	Basic	J	J	J
6.4.14 fo:region-body	Basic	J	J	J
6.4.15 fo:region-before	Extended	J	J	J
6.4.16 fo:region-after	Extended	J	J	J
6.4.17 fo:region-start	Extended	J	J	J
6.4.18 fo:region-end	Extended	J	J	J
6.4.19 fo:flow	Basic	J	J	J
6.4.20 fo:static-content	Extended	J	J	J
6.4.21 fo:title	Extended	J	N	N
6.4.22 fo:flow-map	Extended	J	-	-
6.4.23 fo:flow-assignment	Extended	J	-	-
6.4.24 fo:flow-source-list	Extended	J	-	-
6.4.25 fo:flow-name-specifier	Extended	J	-	-
6.4.26 fo:flow-target-list	Extended	J	-	-
6.4.27 fo:region-name-specifier	Extended	J	-	-

6.5 Block-level Formatting Objects

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
6.5.2 fo:block	Basic	J	J	J
6.5.3 fo:block-container	Extended	J	P	J

6.6 Inline-level Formatting Objects

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
6.6.2 fo:bidi-override	Extended	J	N	J
6.6.3 fo:character	Basic	J	J	J
6.6.4 fo:initial-property-set	Extended	N	N	J
6.6.5 fo:external-graphic	Basic	J	J	J
6.6.6 fo:instream-foreign-object	Extended	J	J	J
6.6.7 fo:inline	Basic	J	J	J
6.6.8 fo:inline-container	Extended	J	N	N
6.6.9 fo:leader	Basic	J	P	J
6.6.10 fo:page-number	Basic	J	J	J
6.6.11 fo:page-number-citation	Extended	J	P	J

6.7 Formatting Objects forTables

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
6.7.2 fo:table-and-caption	Basic	J	N	J
6.7.3 fo:table	Basic	J	P	J
6.7.4 fo:table-column	Basic	J	J	J
6.7.5 fo:table-caption	Extended	J	N	J
6.7.6 fo:table-header	Basic	J	J	J

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
6.7.7 fo:table-footer	Extended	J	J	J
6.7.8 fo:table-body	Basic	J	J	J
6.7.9 fo:table-row	Basic	J	J	J
6.7.10 fo:table-cell	Basic	J	J	J

6.8 List FormattingObjects

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
6.8.2 fo:list-block	Basic	J	J	J
6.8.3 fo:list-item	Basic	J	J	J
6.8.4 fo:list-item-body	Basic	J	J	J
6.8.5 fo:list-item-label	Extended	J	J	J

6.9 Dynamic Effects: Link and Multi FormattingObjects

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
6.9.2 fo:basic-link	Extended	J	J	J
6.9.3 fo:multi-switch	Extended	N	N	N
6.9.4 fo:multi-case	Basic	J	N	N
6.9.5 fo:multi-toggle	Extended	N	N	N
6.9.6 fo:multi-properties	Extended	N	N	N
6.9.7 fo:multi-property-set	Extended	N	N	N

6.10 Formatting Objects for Indexing

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
6.10.2 fo:index-page-number	Extended	J	-	-
6.10.3 fo:index-page-number-suffix	Extended	J	-	-
6.10.4 fo:index-range-begin	Extended	J	-	-
6.10.5 fo:index-range-end	Extended	J	-	-
6.10.6 fo:index-key-reference	Extended	J	-	-
6.10.7 fo:index-page-citation-list	Extended	J	-	-
6.10.8 fo:index-page-citation-list-seperator	Extended	J	-	-
6.10.9 fo:index-page-citation-range-seperator	Extended	J	-	-

6.11 Formatting Objects for Bookmarks

Name	Conformance Level	Antenna House	FOP	RenderX XEP
	Supported	Supported	Supported	Supported
6.11.2 fo:bookmark-tree	Extended	J	J	-
6.11.3 fo:bookmark	Extended	J	J	-
6.11.4 fo:footnote-body	Extended	J	P	-

6.12 Out-of-Line Formatting Objects

Name	Conformance Level	Antenna House	FOP	RenderX XEP
	Supported	Supported	Supported	Supported
6.12.2 fo:float	Extended	J	N	P
6.12.3 fo:footnote	Extended	J	J	J
6.12.4 fo:footnote-body	Extended	J	J	J

6.13 Other Formatting Objects

Name	Conformance Level	Antenna House	FOP	RenderX XEP
	Supported	Supported	Sup-ported	Supported
6.13.2 fo:change-bar-begin	Extended	J	-	-
6.13.3 fo:change-bar-end	Extended	J	-	-
6.13.4 fo:wrapper	Basic	J	J	J
6.13.5 fo:marker	Extended	J	J	J
6.13.6 fo:retrieve-marker	Extended	J	J	J
6.13.7 fo:retrieve-table-marker	Extended	J	-	-

7 XSL-FO Property Support

7.5Common Accessibility Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.5.1 source-document	Basic	J	N	N
7.5.2 role	Basic	J	N	N

7.6 Common Absolute PositionProperties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.6.1 absolute-position	Complete	J	J	P
7.6.2 top	Extended	J	J	J
7.6.3 right	Extended	J	J	J
7.6.4 bottom	Extended	J	J	J
7.6.5 left	Extended	J	J	J

7.8 Common Border, Padding, and BackgroundProperties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.8.1 background-attachment	Extended	J	N	J
7.8.2 background-color	Basic	J	J	J
7.8.3 background-image	Extended	J	J	J
7.8.4 background-repeat	Extended	J	N	J

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.8.5 background-position-horizontal	Extended	J	P	P
7.8.6 background-position-vertical	Extended	J	P	P
7.8.7 border-before-color	Basic	J	J	J
7.8.8 border-before-style	Basic	J	J	J
7.8.9 border-before-width	Basic	J	J	J
7.8.10 border-after-color	Basic	J	J	J
7.8.11 border-after-style	Basic	J	J	J
7.8.12 border-after-width	Basic	J	J	J
7.8.13 border-start-color	Basic	J	J	J
7.8.14 border-start-style	Basic	J	J	J
7.8.15 border-start-width	Basic	J	J	J
7.8.16 border-end-color	Basic	J	J	J
7.8.17 border-end-style	Basic	J	J	J
7.8.18 border-end-width	Basic	J	J	J
7.8.19 border-top-color	Basic	J	J	J
7.8.20 border-top-style	Basic	J	J	J
7.8.21 border-top-width	Basic	J	J	J
7.8.22 border-bottom-color	Basic	J	J	J
7.8.23 border-bottom-style	Basic	J	J	J
7.8.24 border-bottom-width	Basic	J	J	J

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.8.25 border-left-color	Basic	J	J	J
7.8.26 border-left-style	Basic	J	P	J
7.8.27 border-left-width	Basic	J	J	J
7.8.28 border-right-color	Basic	J	J	J
7.8.29 border-right-style	Basic	J	J	J
7.8.30 border-right-width	Basic	J	J	J
7.8.31 padding-before	Basic	J	J	J
7.8.32 padding-after	Basic	J	J	J
7.8.33 padding-start	Basic	J	J	J
7.8.34 padding-end	Basic	J	J	J
7.8.35 padding-top	Basic	J	J	J
7.8.36 padding-bottom	Basic	J	J	J
7.8.37 padding-left	Basic	J	J	J
7.8.38 padding-right	Basic	J	J	J

7.9 Common Font Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.9.2 font-family	Basic	J	P	J
7.9.3 font-selection-strategy	Complete	J	N	J
7.9.4 font-size	Basic	J	J	J

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.9.5 font-stretch	Extended	J	N	J
7.9.6 font-size-adjust	Extended	J	N	J
7.9.7 font-style	Basic	P	J	J
7.9.8 font-variant	Basic	J	N	N
7.9.9 font-weight	Basic	J	P	J

7.10 Common HyphenationProperties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.10.1 country	Extended	J	J	N
7.10.2 language	Extended	J	J	J
7.10.3 script	Extended	J	N	N
7.10.4 hyphenate	Extended	J	J	J
7.10.5 hyphenation-character	Extended	J	J	J
7.10.6 hyphenation-push-character-count	Extended	J	J	J
7.10.7 hyphenation-remain-character-count	Extended	J	J	J

7.11 Common Margin Properties-Block

Name	Conformance	Antenna House	FOP	RenderX XEP

	Level	Supported	Supported	Supported
7.11.1 margin-top	Basic	J	J	J
7.11.2 margin-bottom	Basic	J	J	J
7.11.3 margin-left	Basic	J	J	J
7.11.4 margin-right	Basic	J	J	J
7.11.5 space-before	Basic	J	P	J
7.11.6 space-after	Basic	J	P	P
7.11.7 start-indent	Basic	J	J	J
7.11.8 end-indent	Basic	J	J	J

7.12 Common MarginProperties-Inline

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.12.1 margin-top	Basic	J	-	-
7.12.2 margin-bottom	Basic	J	-	-
7.12.3 margin-left	Basic	J	-	-
7.12.4 margin-right	Basic	J	-	-
7.12.5 space-end	Basic	J	N	J
7.12.6 space-start	Basic	J	N	J

7.13 Common RelativePosition Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.13.1 top	Extended	J	-	-
7.13.2 right	Extended	J	-	-

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.13.3 bottom	Extended	J	-	-
7.13.4 left	Extended	J	-	-
7.13.5 relative-position	Extended	J	N	N

7.14 Area Alignment Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.14.1 alignment-adjust	Basic	J	J	J
7.14.2 alignment-baseline	Basic	J	J	J
7.14.3 baseline-shift	Basic	J	J	J
7.14.4 display-align	Extended	J	P	J
7.14.5 dominant-baseline	Basic	J	J	J
7.14.6 relative-align	Extended	J	N	P

7.15 Area Dimension Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.15.1 allowed-height-scale	Extended	J	-	-
7.15.2 allowed-width-scale	Extended	J	-	-
7.15.3 block-progression-dimension	Basic	J	J	J

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.15.4 content-height	Extended	J	J	J
7.15.5 content-width	Extended	J	J	J
7.15.6 height	Basic	J	J	J
7.15.7 inline-progression-dimension	Basic	J	J	J
7.15.8 max-height	Complete	J	N	N
7.15.9 max-width	Complete	J	N	N
7.15.10 min-height	Complete	J	N	N
7.15.11 min-width	Complete	J	N	N
7.15.12 scaling	Extended	J	J	J
7.15.13 scaling-method	Extended	N	N	N
7.15.14 width	Basic	J	J	J

7.16 Block and Line-related Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.16.1 hyphenation-keep	Extended	J	N	N
7.16.2 hyphenation-ladder-count	Extended	J	J	N

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.16.3 last-line-end-indent	Extended	J	J	J
7.16.4 line-height	Basic	J	J	J
7.16.5 line-height-shift-adjustment	Extended	J	J	J
7.16.6 line-stacking-strategy	Basic	J	P	J
7.16.7 linefeed-treatment	Extended	J	J	P
7.16.8 white-space-treatment	Extended	J	P	J
7.16.9 text-align	Basic	J	P	P
7.16.10 text-align-last	Extended	J	P	J
7.16.11 text-indent	Basic	J	J	J
7.16.12 white-space-collapse	Extended	J	J	P
7.16.13 wrap-option	Basic	J	J	J

7.17 CharacterProperties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.17.1 character	Basic	J	J	J
7.17.2 letter-spacing	Extended	J	J	J
7.17.3 suppress-at-line-break	Extended	N	N	N
7.17.4 text-decoration	Extended	J	J	J
7.17.5 text-shadow	Extended	N	N	P
7.17.6 text-transform	Extended	J	J	P

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.17.7 treat-as-word-space	Extended	N	N	N
7.17.8 word-spacing	Extended	J	N	J

7.18 Color-related Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.18.1 color	Basic	J	J	J
7.18.2 color-profile-name	Extended	J	N	N
7.18.3 rendering-intent	Extended	N	N	N

7.19 Float-related Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.19.1 clear	Extended	J	N	J
7.19.2 float	Extended	J	N	P
7.19.3 intrusion-displace	Extended	J	N	P

7.20 Keeps and Breaks Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.20.1 break-after	Basic	J	J	J
7.20.2 break-before	Basic	J	J	J
7.20.3 keep-together	Extended	J	P	P

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.20.4 keep-with-next	Basic	J	P	P
7.20.5 keep-with-previous	Basic	J	P	P
7.20.6 orphans	Basic	J	J	J
7.20.7 widows	Basic	J	J	J

7.21 Layout-related Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.21.1 clip	Extended	J	N	N
7.21.2 overflow	Basic	J	J	P
7.21.3 reference-orientation	Extended	J	J	J
7.21.4 span	Extended	J	J	J

7.22 Leader and Rule Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.22.1 leader-alignment	Extended	P	N	N
7.22.2 leader-pattern	Basic	J	J	J
7.22.3 leader-pattern-width	Extended	J	J	J
7.22.4 leader-length	Basic	J	J	J
7.22.5 rule-style	Basic	J	J	J
7.22.6 rule-thickness	Basic	J	J	J

7.23 Properties for Dynamic Effects Formatting Objects

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.23.1 active-state	Extended	N	N	N
7.23.1 auto-restore	Extended	N	N	N
7.23.3 case-name	Extended	N	N	N
7.23.4 case-title	Extended	N	N	N
7.23.5 destination-placement-offset	Extended	N	N	N
7.23.6 external-destination	Extended	J	J	J
7.23.7 indicate-destination	Extended	N	N	N
7.23.8 internal-destination	Extended	J	J	J
7.23.9 show-destination	Extended	J	N	J
7.23.10 starting-state	Extended	N	P	N
7.23.11 switch-to	Extended	N	N	N
7.23.12 target-presentation-context	Extended	N	N	N
7.23.13 target-processing-context	Extended	N	N	N
7.23.14 target-stylesheet	Extended	N	N	N

7.24 Properties for Indexing

Name	Conformance Level	Antenna House	FOP	RenderX XEP

		Supported	Sup-ported	Supported
7.24.1 index-class	Extended	J	-	-
7.24.2 index-key	Extended	J	-	-
7.24.3 page-number-treatment	Extended	J	-	-
7.24.4 merge-ranges-across-index-key-references	Extended	J	-	-
7.24.5 merge-sequential-page-numbers	Extended	J	-	-
7.24.6 merge-pages-across-index-key-references	Extended	J	-	-
7.24.7 ref-index-key	Extended	J	-	-

7.25 Properties for Markers

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.25.1 marker-class-name	Extended	J	J	J
7.25.2 retrieve-boundary-within-table	Extended	J	-	-
7.25.3 retrieve-class-name	Extended	J	J	J
7.25.4 retrieve-position	Extended	J	J	J
7.25.5 retrieve-boundary	Extended	J	J	J
7.25.6 retrieve-position-within-table	Extended	j	-	-

7.26 Properties for Number to String Conversion

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.26.1 format	Basic	J	J	J
7.26.2 grouping-separator	Extended	J	N	N
7.26.3 grouping-size	Extended	J	N	N
7.26.4 letter-value	Basic	P	N	N

7.27 Pagination and ayLayout Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.27.1 blank-or-not-blank	Extended	J	J	J
7.27.2 column-count	Extended	J	J	J
7.27.3 column-gap	Extended	J	J	J
7.27.4 extent	Extended	J	J	J
7.27.5 flow-name	Basic	J	J	J
7.27.6 force-page-count	Extended	J	J	J
7.27.7 initial-page-number	Basic	J	J	J
7.27.8 master-name	Basic	J	J	J
7.27.9 master-reference	Basic	J	J	J
7.27.10 maximum-repeats	Extended	J	J	J
7.27.11 media-usage	Extended	N	N	N

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.27.12 odd-or-even	Extended	J	J	J
7.27.13 page-height	Basic	J	J	J
7.27.14 page-position	Extended	J	P	J
7.27.15 page-width	Basic	J	J	J
7.27.16 precedence	Extended	J	N	J
7.27.17 region-name	Basic	J	J	J
7.27.18 flow-map-name	Extended	J	-	-
7.27.19 flow-map-reference	Extended	J	-	-
7.27.20 flow-name-reference	Extended	J	-	-
7.27.21 region-name-reference	Extended	J	-	-

7.28 Table Properties

Name	Conformance Level	Antenna House	FOP	RenderXXEP
		Supported	Sup-ported	Supported
7.28.1 border-after-precedence	Basic	J	N	J
7.28.2 border-before-precedence	Basic	J	N	J
7.28.3 border-collapse	Extended	J	J	J
7.28.4 border-end-precedence	Basic	J	N	J

Name	Conformance Level	Antenna House	FOP	RenderXXEP
		Supported	Sup-ported	Supported
7.28.5 border-separation	Extended	J	J	J
7.28.6 border-start-precedence	Basic	J	N	J
7.28.7 caption-side	Complete	J	N	P
7.28.8 column-number	Basic	J	J	J
7.28.9 column-width	Basic	J	J	J
7.28.10 empty-cells	Extended	N	N	N
7.28.11 ends-row	Extended	J	J	J
7.28.12 number-columns-repeated	Basic	J	J	J
7.28.13 number-columns-spanned	Basic	J	J	J
7.28.14 number-rows-spanned	Basic	J	J	J
7.28.15 starts-row	Extended	J	J	J
7.28.16 table-layout	Extended	J	N	J
7.28.17 table-omit-footer-at-break	Extended	J	J	N
7.28.18 table-omit-header-at-break	Extended	J	J	J

7.29 Writing-mode-related Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP

		Supported	Sup-ported	Supported
7.29.1 direction	Basic	J	N	J
7.29.2 glyph-orientation-horizontal	Extended	J	N	N
7.29.3 glyph-orientation-vertical	Extended	J	N	N
7.29.4 text-altitude	Extended	J	N	J
7.29.5 text-depth	Extended	J	N	J
7.29.6 unicode-bidi	Extended	J	N	P
7.29.7 writing-mode	Basic	P	N	P

7.30 Miscellaneous Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.30.1 change-bar-class	Extended	J	-	-
7.30.2 change-bar-color	Extended	J	-	-
7.30.3 change-bar-offset	Extended	J	-	-
7.30.4 change-bar-placement	Extended	J	-	-
7.30.5 change-bar-style	Extended	J	-	-
7.30.6 change-bar-width	Extended	J	-	-
7.30.7 content-type	Extended	J	N	J
7.30.8 id	Basic	J	P	J
7.30.9 intrinsic-scale-value	Extended	J	-	-

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Sup-ported	Supported
7.30.10 page-citation-strategy	Extended	J	-	-
7.30.11 provisional-label-separation	Basic	J	J	J
7.30.12 provisional-distance-between-starts	Basic	J	J	J
7.30.13 ref-id	Extended	J	J	J
7.30.14 scale-option	Extended	J	-	-
7.30.15 score-spaces	Extended	N	N	N
7.30.16 src	Basic	J	J	J
7.30.17 visibility	Extended	J	N	N
7.30.18 z-index	Extended	J	N	N

7.31 Shorthand Properties

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.31.1 background	Complete	J	N	J
7.31.2 background-position	Complete	J	J	J
7.31.3 border	Complete	J	J	J
7.31.4 border-bottom	Complete	J	J	J
7.31.5 border-color	Complete	J	J	J
7.31.6 border-left	Complete	J	J	J
7.31.7 border-right	Complete	J	J	J

Name	Conformance Level	Antenna House	FOP	RenderX XEP
		Supported	Supported	Supported
7.31.8 border-style	Complete	J	j	J
7.31.9 border-spacing	Complete	J	J	J
7.31.10 border-top	Complete	J	J	J
7.31.11 border-width	Complete	J	J	J
7.31.12 cue	Complete	N	N	N
7.31.13 font	Complete	J	P	J
7.31.14 margin	Complete	J	J	J
7.31.15 padding	Complete	J	J	J
7.31.16 page-break-after	Complete	J	J	J
7.31.17 page-break-before	Complete	J	J	J
7.31.18 page-break-inside	Complete	J	J	J
7.31.19 pause	Complete	N	N	N
7.31.20 position	Complete	J	J	J
7.31.21 size	Complete	J	N	J
7.31.22 vertical-align	Complete	J	P	J
7.31.23 white-space	Complete	J	J	J
7.31.24 xml:lang	Complete	J	N	N

Tabelle 3 Unterstützung der FO-Standards [13][14][20][21]

Ehrenwörtliche Erklärung

Ich erkläre hiermit ehrenwörtlich, dass ich die vorliegende Diplomarbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommene Gedanken sind als solche kenntlich gemacht.

An der geistigen Herstellung der vorliegenden Diplomarbeit war außer mir niemand beteiligt. Insbesondere habe ich nicht die Hilfe eines Diplomberaters in Anspruch genommen. Dritte haben von mir weder unmittelbar noch mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorliegenden Diplomarbeit stehen.
Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form oder auszugsweise einer Prüfungsbehörde vorgelegt.