

op-Whitepaper

# Elektronische Posteingangsbearbeitung mit Open Source Software

Autor: Dr. Mark-Christoph Müller  
Senior Consultant  
objective partner AG

mark-christoph.mueller@objective-partner.de  
0162 / 2705332

## Einleitung

Fortgeschrittene elektronische Posteingangsbearbeitung, die über einfaches Scannen und Ablegen im Dokumenten-Management-System (DMS) hinausgeht, ist ein klarer Trend: Laut einer aktuellen Studie<sup>1</sup> setzt zwar über die Hälfte der befragten Unternehmen noch keine derartige Technologie für die elektronische Posteingangsbearbeitung ein. Bei über 70% dieser Unternehmen ist eine Einführung aber geplant. In den meisten Fällen umfasst der dabei vorgesehene Funktionsumfang fortgeschrittene Funktionen wie Texterkennung, Dokument-Klassifikation und Datenextraktion zur Übergabe an nachgeordnete ERP-Systeme. Der Markt bietet hierfür eine Vielzahl von leistungsfähigen, hoch optimierten und dabei oft kostspieligen Lösungen an. Open Source Software (OSS) stellt eine mögliche Alternative zu diesen kommerziellen Anwendungen dar. Neben dem Wegfall der Lizenzkosten bietet OSS potentiell noch andere Vorteile.

Der vorliegende Artikel geht der Frage nach, ob für die fortgeschrittene elektronische Posteingangsbearbeitung heute verfügbare OSS eine praxistaugliche Alternative zu kommerziellen Anwendungen darstellt. Besonderes Augenmerk gilt dabei der Texterkennung, die eine kritische Bedeutung für den gesamten Bearbeitungsprozess hat. Bei der Darstellung der OSS-Lösungen werden sowohl komplett integrierte OSS-Anwendungen als auch OSS-Komponenten und Software-Bibliotheken als Grundlage von Eigenentwicklungen betrachtet.

## Vorteile von OSS

Welche Gründe gibt es für ein Unternehmen, den Einsatz von OSS für einen so wichtigen Bereich wie die elektronische Posteingangsbearbeitung zu erwägen? Als die wesentlichen Vorteile von OSS werden, neben der meist kostenlosen Verfügbarkeit, häufig die folgenden Punkte genannt.

1. Durch die Verfügbarkeit des Quellcodes kann die Funktionsweise der Software nachvollzogen und vor allem modifiziert werden. Dadurch kann OSS sehr gut an spezielle Anforderungen eines Unternehmens angepasst werden.
2. Da die Schnittstellen offen liegen, ist auch ein hohes Maß an Integrierbarkeit gegeben.
3. Infolge der unter 1. und 2. genannten Vorteile macht sich ein Unternehmen beim Einsatz von OSS nicht von einem einzelnen Softwareanbieter oder von der Verfügbarkeit von Updates oder Bug Fixes abhängig.
4. Durch eine ausreichend große Entwickler-Community und das dadurch ermöglichte Peer-Reviewing<sup>2</sup> kann zumindest theoretisch bessere, effizientere und sicherere Software mit weniger Fehlern produziert werden. In der Praxis wird jedoch beim Großteil der OSS-Projekte der Programmcode von einem relativ kleinen Team aus Entwicklern produziert. Der Beitrag der Community beschränkt sich hier auf Input in Form von z.B. Bug Reports, Software-Lokalisierungen, Dokumentation und die Erstellung von Plug-Ins.

Ein Beispiel für ein erfolgreiches OSS-Projekt mit einer großen und aktiven Community ist OpenOffice.org<sup>3</sup>, das eine kostenlose Alternative zu kommerziellen Office-Anwendungen für Endanwender darstellt. Ein anderes Beispiel sind die diversen OSS-Projekte der Apache Software Foundation<sup>4</sup>. Die meisten dieser Projekte adressieren stark technikhorientierte Themen wie z.B. Internettechnologien, Tools für die Softwareentwicklung, XML-Technologien oder Datenbanken. Sie bringen hauptsächlich Komponenten und Bibliotheken für die Verwendung durch Softwareentwickler

<sup>1</sup> PENTADOC Radar – Elektronische Posteingangsbearbeitung in der Praxis, Juli 2010.

<sup>2</sup> Gegenseitige Begutachtung des Programmcodes

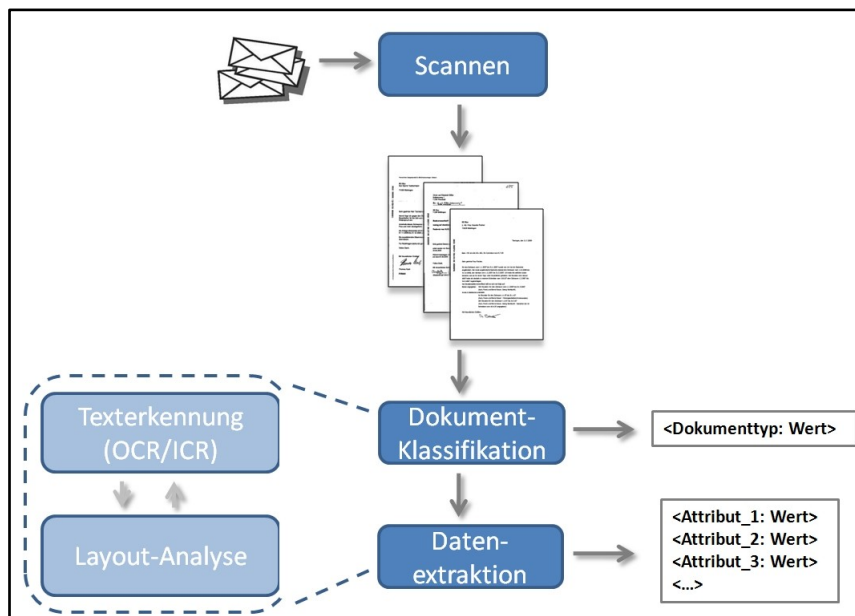
<sup>3</sup> <http://de.openoffice.org>

<sup>4</sup> <http://www.apache.org>

hervor. Sie sind thematisch nicht auf normale Endanwender ausgerichtet, sondern werden eher „von der Community für die Community“ entwickelt.

## Fortgeschrittene elektronische Posteingangsbearbeitung

Im Gegensatz zum einfachen Scannen und Ablegen der Eingangspost bei manueller Bearbeitung strebt die fortgeschrittene elektronische Posteingangsbearbeitung eine weitgehende Automatisierung möglichst vieler Bearbeitungsschritte an. Im Idealfall wird dadurch z.B. für Eingangsrechnungen eine komplette Automatisierung inkl. Buchung im ERP (Dunkelverarbeitung) erreicht. Seitens der Posteingangsbearbeitung sind dafür neben dem Scannen die folgenden weiteren Bearbeitungsschritte erforderlich.



**Bearbeitungsschritte der fortgeschrittenen elektronischen Posteingangsbearbeitung**

Die **Texterkennung** liefert das Rohmaterial für die gesamte weitere Bearbeitung des Eingangsdokuments. Fehler in dieser frühen Bearbeitungsphase können sich negativ auf alle folgenden Schritte auswirken. Durch die Texterkennung wird der Inhalt des gescannten Eingangsdokuments in maschinenlesbaren Text umgewandelt, der dann z.B. zusammen mit dem Dokumentenimage in einer durchsuchbaren PDF-Datei gespeichert wird. OCR (Optical Character Recognition) analysiert dazu die einzelnen Bildpunkte im Dokumentenimage und ordnet die Muster den entsprechenden Buchstaben oder Ziffern zu. Wenn es mehrere mögliche Zuordnungen gibt, können für die Auswahl des korrekten Zeichens ICR-Verfahren verwendet werden. ICR (Intelligent Character Recognition) nutzt dafür Informationsquellen wie Lexika, Stammdatensätze aus der ERP-Datenbank oder Informationen über wahrscheinliche Buchstaben- oder Wortkombinationen. Fortgeschrittene Texterkennung schließt außerdem eine Layout-Analyse mit ein. Bei mehrspaltigen Eingangsdokumenten ist sie erforderlich, um den Textfluss korrekt zu erkennen. Bei Dokumenten mit komplexem Layout (z.B. Tabellen) ist eine präzise Layout-Analyse außerdem eine unabdingbare

Voraussetzung für die korrekte Datenextraktion (s.u.). Im besten Fall liefert die Layout-Analyse Informationen in Form von sog. Bounding Boxes<sup>5</sup> für alle erkannten Wörter.

Bei der **Dokument-Klassifikation** wird das Eingangsdokument automatisch einem bestimmten Dokumenttyp wie z.B. Rechnung, Bestellung oder Adressänderung zugeordnet. Bei formlosen Eingangsdokumenten basiert die Dokument-Klassifikation ausschließlich auf den zuvor erkannten Wörtern. Wenn das Dokument z.B. einen oder mehrere der Ausdrücke „Rechnung“, „zahlbar“ und „auf das Konto“ enthält, handelt es sich höchstwahrscheinlich um eine Rechnung. Bei Formularen oder regelmäßig eingehenden Dokumenten mit dem immer gleichen Layout (wie z.B. Rechnungen von Stammlieferanten) kommen auch Verfahren zum Einsatz, die zusätzlich Grafiken oder Logos im Dokument berücksichtigen. Die Klassifikation erfolgt dabei häufig durch Vergleich mit hinterlegten Beispieldokumenten. Die manuelle Erstellung und Pflege eines Klassifikations-Regelwerks ist grundsätzlich recht zeitaufwendig. Sofern die zu klassifizierenden Dokumenttypen leicht erkennbare und relativ sichere Unterscheidungsmerkmale aufweisen, können damit sehr hohe Klassifikationsraten erzielt werden. Änderungen des Regelwerks, z.B. durch Hinzukommen eines weiteren Dokumenttyps, sind aber aufwendig und können die Klassifikationsleistung des bereits existierenden Regelwerks beeinträchtigen.

Bei der **Datenextraktion** werden Werte für bestimmte Attribute gezielt aus dem Dokument extrahiert. Bei Rechnungen sind dies z.B. Lieferant, Bestellnummer und Rechnungsbetrag, bei Adressänderungen z.B. Kundennummer und neue Postanschrift. Bei der Variante mittels **Freiform-Erkennung** werden Attribut-Werte anhand von Schlüsselwörtern und relativen Positionsangaben identifiziert. Die Bestellnummer steht z.B. häufig rechts von oder direkt unter einem der Schlüsselwörter „Bestellnummer“, „Bestellnr“, „Bestellung“, „Ihr Zeichen“ etc. Die Leistungsfähigkeit dieser Methode hängt davon ab, wie detailliert die Layout-Information für das Dokument ist: Im einfachsten Fall kann lediglich horizontal innerhalb einer Zeile (rechts bzw. links vom Schlüsselwort) gesucht werden. Positionierung in vertikaler Richtung (direkt über bzw. unter dem Schlüsselwort) erfordert detailliertere Layout-Informationen. Bei häufig auftretenden Dokumenttypen mit immer gleichem Aufbau kann auch **maskenbasierte Extraktion** eingesetzt werden. Dabei werden mit Hilfe von Beispieldokumenten und manuell angelegten Schablonen Inhalte gezielt von bestimmten Positionen des Dokuments ausgelesen. Im Rahmen der Datenextraktion werden häufig Stamm- und Bewegungsdaten aus der ERP-Datenbank zur datenbasierten Plausibilisierung der Extraktionsergebnisse herangezogen. Außerdem können so Fehler oder Unsicherheiten aus der Texterkennung teilweise kompensiert werden.

## **OSS für die elektronische Posteingangsbearbeitung**

### **OSS-Texterkennungs-Anwendungen**

Eine der wenigen erstzunehmenden OSS-Texterkennungs-Anwendungen ist **Tesseract**<sup>6</sup>, das ursprünglich kommerziell von der Firma HP entwickelt wurde und seit 2006 als OSS-Projekt bei Google gehostet wird. Tesseract alleine bietet nur Texterkennung ohne Layout-Analyse. Besondere Bedeutung erlangt Tesseract im Zusammenspiel mit **OCRopus**<sup>7</sup>. OCRopus wird seit 2007 ebenfalls bei Google als OSS-Projekt gehostet. OCRopus integriert Tesseract als OCR-Engine und stellt fortgeschrittene Sprach-Modellierung und Layout-Analyse zur Verfügung. Eine Alternative zu

<sup>5</sup> Umschließende Rechtecke, definiert durch X- und Y-Koordinate einer Ecke sowie durch Breite und Höhe.

<sup>6</sup> <http://code.google.com/p/tesseract-ocr/>

<sup>7</sup> <http://code.google.com/p/ocropus/>

Tesseract/OCRopus ist **cuneiform**<sup>8</sup> der russischen Firma Cognitive Technologies, welches seit 2008 ebenfalls als OSS verfügbar ist. cuneiform wurde ursprünglich nur für Windows entwickelt, inzwischen existiert aber auch eine Linux-Portierung<sup>9</sup>. Sowohl OCRopus als auch die Linux-Version von cuneiform können das Erkennungsergebnis im hOCR-Format ausgeben. Dabei handelt es sich um ein erweitertes HTML-Format, welches neben dem erkannten Text auch Layout-Informationen enthält.

Tesseract/OCRopus und cuneiform gelten allgemein als die leistungsfähigsten OSS-Texterkennungs-Anwendungen. Im Jahr 2009<sup>10</sup> ging die Linux-Version von cuneiform (damals in Version 0.5) aus einem direkten Vergleich mit OCRopus (damals in Version 0.3.1) als Sieger hervor. Ein aktuellerer Vergleichstest<sup>11</sup> von cuneiform mit der kommerziellen Texterkennungs-Engine Abby Finereader verweist die OSS-Anwendung allerdings deutlich auf die Plätze. Schwächen zeigen sowohl cuneiform als auch Tesseract/OCRopus bei der Analyse von komplexeren Layouts und v.a. bei Formularen, während die reine Texterkennung zufriedenstellend funktioniert.

### **Integrierte OSS-Anwendungen**

Im OSS-Bereich herrscht kein Mangel an DMS-Projekten.<sup>12</sup> Bei näherem Hinsehen verfügen viele der dort entwickelten Anwendungen jedoch nur über einen geringen Funktionsumfang, und die Unterstützung fortgeschrittener Posteingangsbearbeitungsfunktionen ist meist ohnehin nicht vorgesehen. Demgegenüber ist die Liste der OSS-Anwendungen, die zumindest ansatzweise für die fortgeschrittene elektronische Posteingangsbearbeitung geeignet erscheinen, sehr kurz:

- bitfarm-Archiv DMS Open Source GPL Edition<sup>13</sup>
- KnowledgeTree Community Edition<sup>14</sup>
- Agorum core Open<sup>15</sup>
- Archivista<sup>16</sup>

Mit Ausnahme von KnowledgeTree stammt keine dieser Anwendungen ursprünglich aus der OSS-Community. Vielmehr handelt es sich um freigegebene Versionen rein kommerziell entwickelter Anwendungen, deren Entwicklung weiterhin in den Händen der Hersteller liegt. Die wesentlichen Unterschiede zwischen den OSS-Versionen und den ebenfalls verfügbaren kommerziellen Versionen sind eingeschränkter Funktionsumfang<sup>17</sup> sowie das Fehlen von offiziellem Hersteller-Support. Stattdessen wird Support Community-basiert über Internetforen z.B. bei SourceForge.net geleistet.

Bei **bitfarm-Archiv DMS Open Source GPL Edition** handelt es sich um eine komplette Client-Server-Anwendung inkl. revisionssicherer Archivierung. Neben der OSS-Version gibt es auch eine Enterprise Edition, die nur im Rahmen eines kostenpflichtigen Servicepakets verfügbar und nicht quelloffen ist. Die Veröffentlichung der Quellcodes der OSS-Version wurde bereits mehrfach verschoben und ist bis jetzt (Oktober 2010) noch nicht vollständig erfolgt. Für die Texterkennung ist in der

<sup>8</sup> <http://en.openocr.org/>

<sup>9</sup> <http://launchpad.net/cuneiform-linux>

<sup>10</sup> Kreußel, Peter: „Nachlese“. Linux-Magazin 3/2009.

<sup>11</sup> Kreußel, Peter: „Richtig gelesen?“. Linux-Magazin 07/2010.

<sup>12</sup> Z.B. Suche nach „DMS“ auf SourceForge.net.

<sup>13</sup> <http://www.bitfarm-archiv.de/html/downloads.html>

<sup>14</sup> <http://www.knowledgetree.org>

<sup>15</sup> <http://www.agorum.com/startseite/downloads/agorum-core-open-source-dms-ecm.html>

<sup>16</sup> <https://sourceforge.net/projects/archivista/>

<sup>17</sup> Außer bei Archivista

aktuellen OSS-Version OCRopus in der stark veralteten Version 0.1.1<sup>18</sup> integriert. Die Enterprise Edition bietet demgegenüber eine Schnittstelle zur Integration der kommerziellen Texterkennungs-Anwendung OmniPage. bitfarm-Archiv DMS erlaubt eine einfache regelbasierte Dokument-Klassifikation auf Grundlage der im Dokument erkannten Wörter. Für die Datenextraktion werden neben konventionellen Extraktionsmasken auch einfache Freiform-Extraktionsregeln unterstützt. Diese enthalten unter anderem einen Suchausdruck für die Lokalisation des gesuchten Attributes im Dokumenttext. In der Enterprise Edition werden in diesem Suchausdruck auch reguläre Ausdrücke unterstützt, wodurch die Leistungsfähigkeit der Extraktion deutlich gesteigert wird. In der Extraktionsregel wird außerdem die Position der zu extrahierenden Daten in Relation zum gefundenen Suchausdruck angegeben. Hier wird lediglich die Suche nach rechts und links unterstützt, so dass zeilenübergreifende Datenextraktion nicht möglich ist.

**KnowledgeTree Community Edition** ist ein web-basiertes, in PHP geschriebenes DMS. Kommerziell wird KnowledgeTree in verschiedenen kostenpflichtigen SaaS-Varianten (Software-as-a-Service) und als lokal zu installierende Commercial Edition angeboten. KnowledgeTree Community Edition bringt selber keine Texterkennung mit, es existiert aber offenbar ein Plug-In für die Integration der freien OSS-Texterkennung GOOCR<sup>19</sup>. Die kostenpflichtigen Versionen von KnowledgeTree unterstützen neben anderen fortgeschrittenen Funktionen auch das Capturing, d.h. integriertes Scannen, Texterkennung und Datenextraktion, mit den kommerziellen Anwendungen Kofax Capture und CAPSYS CAPTURE. Die OSS-Version bietet diese Unterstützung nicht.

Bei **Agorum core Open** handelt es sich um ein Dokumenten- und Enterprise-Content-Management-System. Neben der OSS-Version ist auch das kostenpflichtige Agorum core Pro verfügbar. Beide Versionen sind lt. Herstellerangaben im Kern identisch, bestimmte Funktionserweiterungen sind jedoch nur für die kostenpflichtige Version verfügbar. Agorum core Open enthält genau wie KnowledgeTree Community Edition keine eigene Texterkennung. Das kostenpflichtige Agorum core Pro unterstützt demgegenüber die serverseitige Integration von Texterkennung (in diesem Fall die kommerzielle Texterkennungs-Anwendung Abbyy Finereader) über ein Zusatzmodul.

Das webbasierte DMS **Archivista** unterscheidet sich von den bisher erwähnten Systemen dadurch, dass es hier *nur* eine OSS-Version gibt, die stets über den vollen Funktionsumfang verfügt. Die Software kommt in erster Linie in der ArchivistaBox zum Einsatz, welche als DMS-Fertiglösung inkl. Hardware kommerziell vertrieben wird. Kommerzielle Texterkennungs-Anwendungen werden von Archivista offiziell nicht unterstützt. Stattdessen setzt Archivista auch hier auf OSS-Texterkennung in Form von Tesseract und cuneiform.

### **OSS-Komponenten und -Bibliotheken**

Im Vergleich zum Einsatz kompletter, integrierter OSS-Anwendungen ist der Einsatz einzelner Software-Komponenten oder Bibliotheken mit einem deutlich höheren initialen Aufwand verbunden. Bei der Verwendung von Software-Bibliotheken, die lediglich bestimmte Funktionalitäten über eine Programmierschnittstelle (API) zur Verfügung stellen, ist grundsätzlich eigene Programmierung erforderlich. Dies eröffnet jedoch gleichzeitig auch Möglichkeiten, die über ein bloßes Customizing einer vorgegeben Anwendung weit hinausgehen. Angesichts der oben genannten Beschränkungen der OSS-Anwendungen kann bei Vorhandensein des erforderlichen technischen Know-Hows eine Eigenentwicklung auf Grundlage verfügbarer OSS-Komponenten daher eine Alternative sein. Dieser

<sup>18</sup> Stand vom November 2007. Die aktuelle Version von OCRopus ist 0.4.4 (Stand vom März 2010).

<sup>19</sup> <http://www.gocr.de/>

Abschnitt stellt einige dieser OSS-Komponenten vor. Der Bereich OSS-Texterkennung wurde bereits weiter oben behandelt.

Wie oben dargestellt, kann **Dokument-Klassifikation** durch die Anwendung von Regeln geleistet werden. Für den Einsatz von manuell erstellten Regeln existieren einige z.B. auf Java basierende OSS-Regel-Engines, welche in eigene Anwendungen integriert werden können. Eine der mächtigeren Regel-Engines ist **Drools Expert** aus dem JBoss-Projekt Drools<sup>20</sup>. Drools Expert erlaubt die Formulierung von beliebig komplexen Regelwerken. Wegen der völligen Gestaltungsfreiheit sind diese Regeln denen von z.B. bitfarm-Archiv bei weitem überlegen, dafür ist der technische Aufwand natürlich auch ungleich höher.

Eine Alternative zur Dokument-Klassifikation mit einem manuell erstellten Regelwerk ist die automatische Klassifikation mit Methoden des **Text Mining** bzw. **Data Mining**. Derartige Verfahren, bei denen maschinelles Lernen zum Einsatz kommt, werden auch von vielen kommerziellen Anwendungen angeboten. Die Erstellung eines automatischen Dokument-Klassifizierers erfolgt in einer Trainingsphase, in der der Klassifizierer eine vorgegebene Menge von Trainingsdokumenten analysiert. Jedes Trainingsdokument muss dazu vorab manuell mit seinem korrekten Dokumenttyp versehen worden sein. Der Trainingsvorgang läuft dann selbständig ab: Die für die Klassifikation wichtigen Dokumenteigenschaften (meist die enthaltenen Wörter) sowie die Abhängigkeiten zwischen ihnen werden ohne Eingreifen eines menschlichen Experten identifiziert. Der so trainierte Klassifizierer kann dann zur Bestimmung des Dokumenttyps neuer Dokumente verwendet werden. Durch maschinelles Lernen können durchaus leistungsfähige Dokument-Klassifizierer erstellt werden, wenn eine ausreichend große und repräsentative Menge an Trainingsdokumenten verfügbar ist. Im laufenden Betrieb kann eine ständige Anpassung des Klassifizierers an neu hinzukommende oder sich ändernde Dokumenttypen durch Pflege eines Pools von Trainingsdokumenten und gegebenenfalls erneutes Trainieren des Klassifizierers erreicht werden.

Als Teilgebiet der Künstlichen Intelligenz (KI) ist das maschinelle Lernen ein sehr aktives Forschungsgebiet an vielen Universitäten. Da es im akademischen Bereich üblich ist, Software als OSS verfügbar zu machen, stehen für das maschinelle Lernen zahlreiche OSS-Anwendungen und -Bibliotheken zur Verfügung, die für die Entwicklung von Dokument-Klassifizierern eingesetzt werden können. **WEKA**<sup>21</sup> ist schon seit 1999 als OSS verfügbar und wurde seitdem ständig weiterentwickelt. Seit 2010 wird es von der Business Intelligence-Firma Pentaho als *Pentaho Data Mining Community Edition*<sup>22</sup> weitergeführt. WEKA umfasst eine ganze Reihe von Algorithmen für das Trainieren von Regeln, Entscheidungsbäumen oder rein statistischen Klassifizierern. Einmal trainierte Klassifizierer können gespeichert und in eigene Java-Anwendungen integriert werden. Eine weitere Data Mining-Anwendung aus dem OSS-Bereich ist **RapidMiner**<sup>23</sup>. RapidMiner ist an WEKA angelehnt und stellt funktionale Erweiterungen sowie eine verbesserte Benutzeroberfläche zur Verfügung. Außer in der frei verfügbaren Community Edition ist es auch in verschiedenen kostenpflichtigen Enterprise Editions verfügbar, die sich hauptsächlich hinsichtlich Support und Wartung unterscheiden.

Anders als für die Dokument-Klassifikation existieren für die **Datenextraktion** aus Dokumenten keine Standard-Anwendungen. Die Datenextraktion kann im Prinzip jedoch ähnlich wie die Dokument-Klassifikation durch manuelle Extraktions-Regeln bewerkstelligt werden. Die völlige Gestaltungsfrei-

---

<sup>20</sup> <http://jboss.org/drools>

<sup>21</sup> <http://www.cs.waikato.ac.nz/~ml/>

<sup>22</sup> <http://weka.pentaho.com>

<sup>23</sup> <http://www.rapidminer.com>

heit erlaubt auch hier wieder die Formulierung von sehr leistungsfähigen Regeln, die beliebige Informationen berücksichtigen können. Durch Anbindung von ERP-Datenbanken ist dabei z.B. auch die Unterstützung der Extraktion durch datenbasierte Plausibilisierung möglich.

## **Fazit**

Dieser Artikel ging der Frage nach, ob für die fortgeschrittene elektronische Posteingangsbearbeitung heute verfügbare Open Source Software eine praxistaugliche Alternative zu kommerziellen Anwendungen darstellt. Als funktionale Anforderungen für fortgeschrittene elektronische Posteingangsbearbeitung wurden dabei Texterkennung, Dokument-Klassifikation und Datenextraktion angenommen.

Bereits die erste dieser Anforderungen erwies sich als kritisch: Texterkennung und dabei besonders die Layout-Analyse sind ein Flaschenhals der fortgeschrittenen elektronischen Posteingangsbearbeitung. Häufig vorkommende Eingangsdokumente mit hohem Automatisierungspotential - wie z.B. Rechnungen - weisen in der Regel ein komplexes Layout mit Tabellen o.ä. auf. Für die akkurate Datenextraktion aus diesen Dokumenten ist daher eine leistungsfähige und präzise Layout-Analyse unabdingbar. Gerade in der Layout-Analyse aber hinken heute verfügbare OSS-Texterkennungs-Anwendungen den kommerziellen Konkurrenten noch hinterher. Nur wenn hauptsächlich einfach strukturierte Eingangsdokumente verarbeitet werden sollen, sinken die Anforderungen an die Layout-Analyse, und die OSS-Texterkennungs-Anwendungen sind durchaus konkurrenzfähig.

Von den betrachteten integrierten OSS-Anwendungen unterstützt nur bitfarm-Archiv DMS zumindest rudimentär alle für die fortgeschrittene elektronische Posteingangsbearbeitung erforderlichen Funktionen. Die Leistungsfähigkeit der einzelnen Funktionen ist - auch in der kostenpflichtigen Enterprise Edition - jedoch mit denen gängiger kommerzieller Anwendungen nicht zu vergleichen. Die OSS-Version hat darüber hinaus noch weitere funktionale Beschränkungen. Systematische funktionale Beschränkungen der OSS- gegenüber den jeweiligen kommerziellen Versionen finden sich auch bei KnowledgeTree und Agorum core. Bei keiner der OSS-Anwendungen besteht darüber hinaus die Möglichkeit, leistungsfähigere kommerzielle Texterkennungsanwendungen direkt zu integrieren. Diese kritische Erweiterbarkeit ist grundsätzlich den kommerziellen Versionen (mit Ausnahme von Archivista) vorbehalten. Hinter dem Vorgehen der Hersteller, auf diese Weise die Leistungsfähigkeit der frei verfügbaren OSS-Versionen bewusst zu limitieren, kann die durchaus gerechtfertigte Absicht gesehen werden, durch Software- bzw. Beratungsverkauf Geld zu verdienen. In der Praxis reduziert es den Nutzwert der angebotenen OSS-Versionen allerdings erheblich, zumindest sobald fortgeschrittene Funktionalitäten eingesetzt oder auch nur evaluiert werden sollen.

Eine mögliche Alternative zum Einsatz einer integrierten OSS-Anwendung für die fortgeschrittene elektronische Posteingangsbearbeitung ist die Eigenentwicklung einer derartigen Anwendung. Für die Anforderungen dieser Aufgabe stehen einige ausgereifte, mehr oder weniger spezifische OSS-Komponenten zur Verfügung. Dies sind OSS-Regel-Engines für die Dokument-Klassifikation sowie für die Datenextraktion, und OSS-Text Mining-Anwendungen für die Dokument-Klassifikation. Die OSS-Texterkennung ist natürlich auch hier ein Problem: Wenn Dokumente mit komplexerem Layout verarbeitet werden sollen, kann ihre Leistungsfähigkeit für den beabsichtigten Einsatz eventuell nicht ausreichend sein. Bei einer Eigenentwicklung besteht in diesem Fall aber die Möglichkeit, hier den OSS-Pfad zu verlassen und für diese Funktionalität eine kommerzielle Texterkennungs-Anwendung zu integrieren.

Grundsätzlich stellt eine Eigenentwicklung einen erheblichen Aufwand dar. Wenn der volle Funktionsumfang der fortgeschrittenen elektronischen Posteingangsbearbeitung selbst entwickelt werden soll,

macht sie unter wirtschaftlichen Gesichtspunkten wahrscheinlich keinen Sinn. In der Regel wird hier der Rückgriff auf eine kommerzielle, nicht aus dem OSS-Bereich stammende Anwendung günstiger sein. Anders sieht dies aus, wenn z.B. nur Teilfunktionalitäten benötigt werden. Dann kann die „Make-or-Buy“-Entscheidung auch zugunsten von „Make“ ausfallen.

Abschließend ist noch die Frage interessant, inwieweit die hier betrachtete Software tatsächlich die in der Einleitung genannten, mit OSS üblicherweise verbundenen Vorteile aufweist. Für die OSS-Texterkennung-Anwendungen, die Text Mining-Anwendungen sowie die Regel-Engines bietet die Verfügbarkeit des Quellcodes für den Entwickler bzw. den Systemintegrator wohl keinen Mehrwert, denn wegen ihres hohen Grades an Spezialisierung sind diese Komponenten für eigene Optimierungen und Anpassungen an unternehmensspezifische Bedürfnisse wohl kaum zugänglich. Anders sieht es mit den OSS-Vorteilen hinsichtlich der Code-Qualität aus: Dadurch, dass z.B. WEKA schon seit vielen Jahren als OSS verfügbar ist und eine weite Verbreitung und eine große Entwickler- bzw. Anwender-Community hat, hat das Projekt eine hohen Reifegrad erreicht.

Der Quellcode der OSS-Version von bitfarm-Archiv DMS ist noch gar nicht komplett veröffentlicht. Die Beteiligung der Anwender-Community beschränkt sich hier bisher auf die Bereitstellung von Diskussionsforen, die von Mitarbeitern von bitfarm-Archiv betreut werden. Der positive Effekt auf die Qualität der Anwendung, der sich durch die Veröffentlichung des Quellcodes und seine Weiterentwicklung durch die Community ergibt, konnte sich hier also noch nicht einstellen. KnowledgeTree und Agorum core unterstützen Verbesserungen und Erweiterungen aus der Community, indem sie entsprechende web-basierte Schnittstellen zum Zugriff auf den Quellcode bereitstellen. Es ist jedoch davon auszugehen, dass die Entscheidung über die Integration einer Erweiterung aus der Community gegen die kommerziellen Interessen des jeweiligen Herstellers abgewogen wird. Die OSS-Vorteile der Anpassbarkeit und der guten Integrierbarkeit können aber zumindest die OSS-Anwendungen, deren Quellcode komplett veröffentlicht ist, voll ausspielen. In dieser Hinsicht sind sie kommerziellen, nicht quelloffenen Anwendungen klar überlegen. Allerdings gilt dies natürlich nur, wenn tatsächlich die jeweilige OSS-Version produktiv eingesetzt wird oder aber die OSS- und die kommerzielle Version über dieselbe Codebasis verfügen.