Preservation Planning
Thesenpapier: Möglichkeiten und Grenzen der Formaterkennung

Einleitung

Die in der Archivwelt verbreitete Formaterkennung mit DROID basierend auf der Format Registry PRONOM vermag gerade bei für die Archivierung wichtigen Formaten (PDF, TIFF, WAVE) keine oder keine genügend informative Resultate zu liefern. Dieses Dokument analysiert, woran das liegt und wie dem Problem beizukommen ist.

Ein historischer Abriss

In der Frühzeit der Datenverarbeitung waren Programm und Daten untrennbar miteinander verbunden: Eine Lochkartenserie verband Ausführungsanweisungen und zu verarbeitende Daten. Mit der Entwicklung eigentlicher Betriebssysteme wie VSM und UNIX in den 70er Jahren gewinnt aber die klare Zuordnung von Programm und Datendatei an Bedeutung. In erster Ordnung muss das Betriebssystem sicher erkennen können, welches ausführbare Programme sind, und zweitens muss der Benutzer wissen, welche Daten durch welche Programme interpretiert werden können.

Erst mit dem Betriebssystemen von Microsoft wird die uns heute vertraute Lösung mit Dateiendung als Formatkennzeichnung eingeführt.[1]

Magic Number

Grundlage der Formaterkennung ist seit der Frühzeit der Informatik das Konzept der *magic number.* Das heisst, eine bestimmte Bytesequenz, eben die *magic number* am Anfang der Datei, verweist auf den Dateityp. Die *magic number* ist entweder betriebssystemabhängig wie bei ausführbaren Programmdateien oder unterliegt einer Konvention. Ein klassisches Beispiel ist das ZIP-Format: Eine ZIP-Datei beginnt mit den vier Bytes PK\x03\x04 und kann so auf einfache Art erkannt werden.

Das Unix-Programm „file“

Aufbauend auf dem Konzept der *magic number* wurde nicht nur auf UNIX Systemen schon relativ früh das Programm *„file“* zur Formaterkennung entwickelt. Um flexibel zu sein und auch neue Dateiformate erkennen zu können, liest *„file“* eine sogenannte *magic number* Datei, eine einfache Datenbank in einer CSV-artigen Struktur, die eine Verbindung zwischen *magic number* und Dateiformat hergestellt. Die *magic number* Datei erlaubt auch komplexere Anforderungen an das zu suchende Bitmuster zu stellen; so beginnt eine JPEG Datei mit \xFF\xD8 und muss mit \xFF\xD9 enden.

Der sequenzielle Aufbau der *magic number* Datei erlaubt einen Vorrang von einzelnen Regeln über nachfolgende Regeln. Damit wird das Formulieren dieser Regeln in der richtigen Reihenfolge, auch weil die Syntax dieser Regeln nicht gerade selbsterklärend ist, schon bald zu einem eigentlichen Kunststück.

Am Rande sei noch erwähnt, dass die *magic number* Datei für die Anwendung in *„file“* in einem Entscheidungsbaum (*B-tree*) kompiliert werden kann, was eine sehr schnelle Formaterkennung erlaubt.

Der Mechanismus ist nach wie vor bei Webservern, Browsern und E-Mail-Programmen die übliche Art, wie der *MIME type* einer Datei erkannt und nach dem entschieden wird, mit welchem Programm eine heruntergeladene Datei angezeigt werden soll.

MIME type

Nun reicht es in vielen Fällen nicht, nur das Format einer Datei zu erkennen, sondern es ist auch notwendig, ein Format eindeutig zu bezeichnen, um den Austausch von Dateien zwischen verschiedenen Akteuren und Programmen zu gewährleisten. Notwendig dazu ist ein sogenannte *Format Registry*, eine Datenbank, wo alle Formate hinreichend beschrieben und mit einem eindeutigen Bezeichner versehen sind.

MIME (Multipurpose Internet Mail Extensions) war ein erster Versuch, für Internet und E-Mail die Formate zu registrieren und in Formatklassen einzuteilen und damit die Zuordnung zu Programmen, die eine ganze Formatklasse darstellen können, möglich zu machen. *MIME* ist heute in mehreren RFCs[2] spezifiziert und definiert für die im Internet gängigen Formate einen Haupttyp und darin einen spezifischen *MIME type* für jedes Format, z.B. *image/jpeg* oder *application/pdf.*[3]

Bei *MIME* handelt es sich wie bei so vielem im Internet um eine Konvention und nicht um eine streng standardisierte Lösung. Insofern besteht nicht der Anspruch, dass es für jedes Format einen *MIME type* gibt. Die Granularität *MIME type* entspricht etwa der Formatbezeichnung durch Dateinamen *Extension*.

---

[1] Apple verfolgt weiterhin eine andere Lösung: eine Datei besteht aus *data fork*, der eigentlichen Datei und *resource fork*, dem Verweis auf das mit der Datei verbundene Programm.

[2] RFC 5322, RFC 2045, vor allem aber RFC 2046: http://tools.ietf.org/html/rfc2046

[3] *Official list* of all *MIME types* assigned by the IANA (Internet Assigned Number Authority) http://www.iana.org/assignments/media-types/media-types.xhtml

Die Format Registry PRONOM

Darum sind auf Drängen der Gedächtnisinstitutionen verschiedene Versuche unternommen worden, eine umfassende internationale Format Registry zu etablieren, etwa die *Global Digital Format Registry (GDFR)* der *Harvard Library*[4] oder die *Unified Digital Format Registry (UDFR)* der *University of California*[5]. Die Bezeichnungen erläutern hier schon das Programm.

Heute etabliert ist die *technical registry PRONOM* des englischen *National Archives*[6]. PRONOM ist eine Datenbank, die zu jedem Dateiformat die wichtigsten Informationen, wie Name, Version, Formatfamilie, Beschreibung etc. enthält, dazu einen diesem Format zugeordneten Unique Identifier, die PUID[7] (vgl. z.B. die Angaben zu JPEG[8]).

Zusätzlich verbindet PRONOM diese Informationen mit einer technischen Signatur[9], welche das Format eindeutig beschreiben soll. Im Falle von JPEG ist dies Offset 0 „FFD8FFE0{2}4A464946000101(00|01|02)“ und MaxOffset „FFD9“. Wir sehen hier, wie das Konzept der *magic number* in einer etwas erweiterten Form[10] in PRONOM übernommen wurde (siehe JPEG weiter oben).

Formaterkennung mit DROID

Die englischen *National Archives* stellen zu PRONOM auch ein Formaterkennungsprogramm namens DROID zur Verfügung, das mit Hilfe der technischen Signatur die Dateiformate erkennen kann. Dazu können alle Signaturen in Form eines *DROID signature files*[11] aus der Datenbank in XML Format exportiert werden. Die Signaturen selber sind als *pseudo Regular Expression*[12] abgebildet. Es gibt neben DROID auch andere Programme[13], die eine Formaterkennung mit Hilfe der PRONOM Datenbank und dem entsprechenden *Signature file* durchführen.

Grenzen der heutigen Formaterkennung

Die Grenzen der oben skizzierten, auf dem Konzept der *magic number* basierenden heutigen Formaterkennung zeigt zum Beispiel das Bildformat TIFF[14] auf, bei dem die Formaterkennung auf der Basis *magic number*, auch wenn sie mit den Möglichkeiten des *pattern matching* (Mustervergleich mit regulären Ausdrücken) erweitert ist, weder die Version noch die für Archive besonders wichtige *baseline* Ausprägung erkennen kann[15]. Die Informationen zu Version, Farbraum, Kompression etc. sind bei TIFF in einer verketteten Liste von Tags abgelegt; nur ein Lesen dieser Tag-Liste fördert die richtigen Informationen zu Tag. Ein reines Suchen nach Bitmustern in der ganzen Datei ist zeitraubend und in diesem Falle nicht zielführend. Der Grund dafür ist, dass *pattern matching* keine konditionalen Abhängigkeiten, wie sie die Grundlage von strukturierten Metadaten ist, abbilden kann.

Zusammengefasst: Erstens sehen wir, dass ein Format nicht nur in einer Ausprägung vorliegt, sondern in der Regel in Versionen und Varianten; zweitens gibt es offenbar Formate, denen mit einer einfacher Suche nach Bitmustern nicht beizukommen ist, weil zwischen den einzelnen Metadaten strukturelle Abhängigkeiten bestehen.

Versionen und Varianten

Ein Format kann nicht nur in einer Ausprägung vorliegen, sondern ein Format besteht in der Regel aus einer Menge von Formatgenerationen. In der Regel werden diese Versionen des Formats genannt. Dazu kommen Varianten, die sich aus speziellen Anwendungszwecken, verwendeten Komprimierungsalgorithmen und vielem mehr ergeben können. Schon ein relativ einfaches Format wie JPEG kennt in PRONOM drei Versionen (1.00, 1.01 und 1.02) und drei Varianten (JPEG-LS, Baseline und JTIP). Bei TIFF gibt es sieben Versionen und durch die mögliche Kombination von Komprimierungsalgorithmen, Farbraumbeschreibung etc. eine viel

---

[4] GDFR http://library.harvard.edu/preservation/digital-preservation_gdfr.html

[5] UDFR http://www.udfr.org/

[6] PRONOM http://www.nationalarchives.gov.uk/PRONOM/Default.aspx

[7] PUID (PRONOM Unique Identifier), andere Identifier *MIME* und *Apple Uniform Type Identifier* werden ebenfalls aufgelistet.

[8] http://www.nationalarchives.gov.uk/PRONOM/Format/ proForMatSearch.aspx?status=detailReport&id=667

[9] http://www.nationalarchives.gov.uk/PRONOM/Format/proFormatSearch.aspx?status=detailReport&id=668&strPageToDisplay=signatures

[10] Es sind neben reinen Bitmustern auch einfache reguläre Ausdrücke möglich.

[11] Die aktuelle Version ist *DROID_SignatureFile_V72.xml*: http://www.nationalarchives.gov.uk/documents/DROID_SignatureFile_V72.xml

[12] Ein regulärer Ausdruck ist eine Zeichenkette, die der Beschreibung von Mengen von Zeichenketten mit Hilfe bestimmter syntaktischer Regeln dient, siehe: http://de.wikipedia.org/wiki/Regulärer_Ausdruck

[13] FIDO - Format Identification for Digital Objects: http://www.openplanetsfoundation.org/software/fido

[14] Tagged Image File Format

[15] National Archives definiert eine PUID für alle TIFF Varianten (fmt/353) und schreibt dazu *„PUID created for the TIFF format in response to the difficulties we have been having with multiple identification of the format and a consensus on a new interpretation of the standard from within The National Archives and outside with external stakeholders.”* http://www.nationalarchives.gov.uk/PRONOM/Format/proFormatSearch.aspx?status=detailReport&id=1099

grössere Zahl von Varianten; dazu kommt eine Untermenge von Varianten, die als *baseline* TIFF bezeichnet und in der Regel als Format für die Archivierung empfohlen wird.[16]

Strukturierte Metadaten

Komplexere Formate enthalten in der Regel strukturierte Metadaten in Form von Tabellen (sicher das Einfachste), verlinkten Listen oder XML-codierten Metadatenblöcken. Ohne Kenntnis ihrer Struktur diese Metadaten lesen zu wollen, um darin Formatversion oder Variante zu erkennen, ist mehr oder weniger hoffnungslos.

Containerformate

Klassische Containerformate wie WAVE oder MOV bestehen aus einem Container für Metadaten und einem sogenannten *payload* Teil, der die eigentlichen Daten enthält. Das Erkennen des Containerformates ohne Klarheit über das Format des eingebetteten Datenstroms ist von relativ bescheidenem Wert.

XML-basierte Dateiformate

Eine ganze Reihe neuerer Dateiformate sind XML-basiert. Diese bestehen in der Regel aus mehreren XML-Dateien, die auf spezifische Art in einem komprimierten ZIP-Container gespeichert sind (zum Beispiele ODF, OOXML, EPUB etc.). Hier kann erst nach dem Öffnen der ZIP-Datei eine seriöse Formaterkennung vorgenommen werden.

Schlussfolgerung

Wir können zwei Schlussfolgerungen ziehen.

Erstens die naheliegende: Gewissen Formaten, nämlich Formaten mit strukturierten Metadaten, Containerformate und XML-basierte Dateiformaten, ist schwierig oder unmöglich mit der klassischen Suche nach einem Bitmuster beizukommen. Das Problem wird verschärft dadurch, dass es sich bei diesen Typen von Formaten gerade um aktuelle Formate, um für die Archivierung wichtige Formate (wie PDF, TIFF, WAVE, etc.) und um Formate mit eine grossen zukünftigen Potenzial handelt, die keinesfalls einfach übergangen werden können. Es ist darum längerfristig unumgänglich, die Formaterkennung auf neue Erkennungsmethoden abzustützen. Das kann auch die Folgerung nahelegen, dass die Formaterkennung mit einem einzigen Ansatz nicht möglich ist, sondern dass z.B. für JPEG der *magic number* Ansatz verwendet werden kann, für TIFF ein echtes Auslesen der Tags notwendig ist und bei komprimierten ZIP-Containern ein Öffnen des Containers der erste Schritt sein muss.

Zweitens muss die Format Registry unabhängig von der Qualität der Formaterkennung die Formate in der gewünschten und aus archivischer Sicht notwendigen Granularität beschreiben, d.h. unabhängig davon, ob zum Format oder zur Formatversion eine *magic number* oder eine *Signatur* vorliegt. Die Aufgabe der Softwareentwicklung ist es dann, für das entsprechende Format und/oder die Formatfamilie eine geeigneten Erkennungsalgorithmus zu entwickeln.

[16] Siehe Empfehlungen der KOST zu TIFF
http://kost-ceco.ch/cms/index.php?id=238,423,0,0,1,0