EPUB for archival preservation

Johan van der Knijff
KB/ National Library of the Netherlands
20 July 2012
Version for external distribution

Table of Contents

1 Introduction

1.1 Background and scope of this report

Over the last few years, the *EPUB* format has become increasingly popular in the consumer market. A number of publishers have indicated their wish to use *EPUB* for supplying their electronic publications to the KB. In response to this, the KB's Departments of Collection and Collection Care requested an initial study to investigate the suitability of the format for archival preservation. The main questions were:

- What are the main characteristics of *EPUB*?
- What functionality does *EPUB* provide, and is this sufficient for representing e.g. content with sophisticated layout and typography requirements?
- How well is the format supported by software tools that are used in (pre-)ingest workflows?
- How suitable is the format for archival preservation? What are the main risks?

In this report I will try to answer these questions as well as possible.

1.2 Note on preservation objectives

The preservation of (digital) objects is typically driven by specific levels and objectives. For instance, one may only be interested in preserving the original bits, the text that is represented by these bits, or, at the highest level, the text with its original layout and appearance, including graphics. Since *EPUB* is essentially a *reflowable* document format, concepts such as "original look and feel" do not apply as easily as, for example, a fixed-layout formats such as *PDF*. Chapter 5 discusses this in more detail. However, it is important to point out here that this report doesn't make any assumptions about the specific preservation objectives, even though it does highlight the main areas of concern.

1.3 Note on terminology

The *EPUB* format is defined by a set of separate specifications. The naming of these specifications (as well as a lot of other *EPUB*-related terminology) was changed between *EPUB* 2 and 3. In an attempt to keep any confusion to a minimum, I decided to stick to the terminology that is used in *EPUB* 3 wherever possible. The most important *EPUB* 2 equivalents are referred to in footnotes.

1.4 Outline

Chapter 2 gives a brief introduction to *EPUB*. It presents a simple example that illustrates the general structure of an *EPUB* file. Chapters 3 and 4 go more in-depth, focusing on *Core Media Types* and the *Package* and *Navigation* Documents. Chapter 5 discusses the functionality that is provided by *EPUB*. In particular, it covers layout and appearance, and the support for multimedia and scripting. For each of these aspects it also explains the differences that exist between versions 2 and 3 of the format. Processing *EPUB* in an

operational (pre-)ingest workflow requires software tools that are able to provide information about each file object (characterisation). Chapter 6 reviews the main available tools. It also includes some tests on 3 data sets. Chapter 7 reviews the overall suitability of *EPUB* as a preservation format. This is done by evaluating the format against a set of widely used criteria (adapted from *The National Archives* and *Library of Congress*). Finally, Chapter 8 wraps up the main conclusions, and provides some recommendations.

1.5 Acknowledgements

Barbara Sierman is thanked for providing feedback on an early draft of this report. Thanks go out to Thomas Ledoux (Bibliothèque Nationale de France), Misty De Meo (Canadian Museum for Human Rights), Peter May and Andy Jackson (British Library) for their comments to the June 2012 version. Their input has resulted in a number of improvements in the current version.

2 The EPUB format: an overview

2.1 What is EPUB?

EPUB is a file format for digital publications and documents. It has its origins in the *Open eBook* format, a legacy format that was released by the company SoftBook Press, Inc. in 1999 [1]. Further development under the umbrella of the *International Digital Publishing Forum* [2] resulted in the first release (2.0) of *EPUB* in 2007. By then it had also become an official standard. This was followed by a minor revision (2.0.1) in 2010, and a major revision (3.0) in 2011. The current (October 2011) version is 3.0.

2.2 Identifiers

As of May 2012, *EPUB* does not have a registered Internet Media (*MIME*) Type [3], although the unofficial *application/epub+zip* is sometimes used [4]. Also, *EPUB* is not included in the *PRONOM* registry, and does not have a Pronom Unique Identifier (*PUID*).

2.3 EPUB in a nutshell

Before going into any details, it is helpful to first outline the general structure of the format. Conceptually, an *EPUB* file is just an ordinary *ZIP* [5] archive. Inside the archive, the contents of a publication are (in the simplest case) represented as one or more *XHTML* files, which may be organised in one or more directories. *CSS* files (Cascading Style Sheets) are used to define layout and formatting. In addition, a number of *XML* files provide various kinds of (mostly structural) metadata. At a very basic level, an EPUB file could be described as "a webpage inside a *ZIP* file", although this is a bit of an oversimplification. Unsurprisingly, the *EPUB* format is largely based on existing web standards.

2.4 Specifications

Version 3, which is the most recent version of the *EPUB* format, is defined by a set of four separate specifications: [1]

- **EPUB Publications** [6]. This defines the various components that make up a publication, and how these components are tied together. [2]
- **EPUB Content Documents** [7] specifies how *content* is represented. More specifically, it defines profiles of *XHTML*, *SVG* and *CSS*.[3]
- **EPUB Open Container Format** [8]. This defines how the components of an *EPUB* publication are encapsulated into a single *ZIP* file [4]

---

[1]Note that this report follows the terminology of *EPUB* 3 here. Confusingly the *EPUB* 2 equivalents of these specifications are named differently; see footnotes below.

[2]In *EPUB* 2 this is called "Open Packaging Format" (*OPF*).

[3]In *EPUB* 2 this is called "Open Publication Structure" (*OPS*).

- **EPUB Media Overlays** [9] defines how text and audio (e.g. using text-to-speech technology, or pre-recorded audio clips) are synchronised. [5]

2.5 A note on *Publication Resources*

The *EPUB* specification uses the term *Publication Resource* to indicate *any* individual component that is part of a publication. For example, an *XHTML* file that contains one chapter of a book is a Publication Resource, and so is the *Package Document* (described below) that contains bibliographic and structural metadata.

2.6 Example

The general structure of *EPUB* can be best shown using a simple example. Figure 1 below shows the contents of a (fairly simple) *EPUB* 3 file.

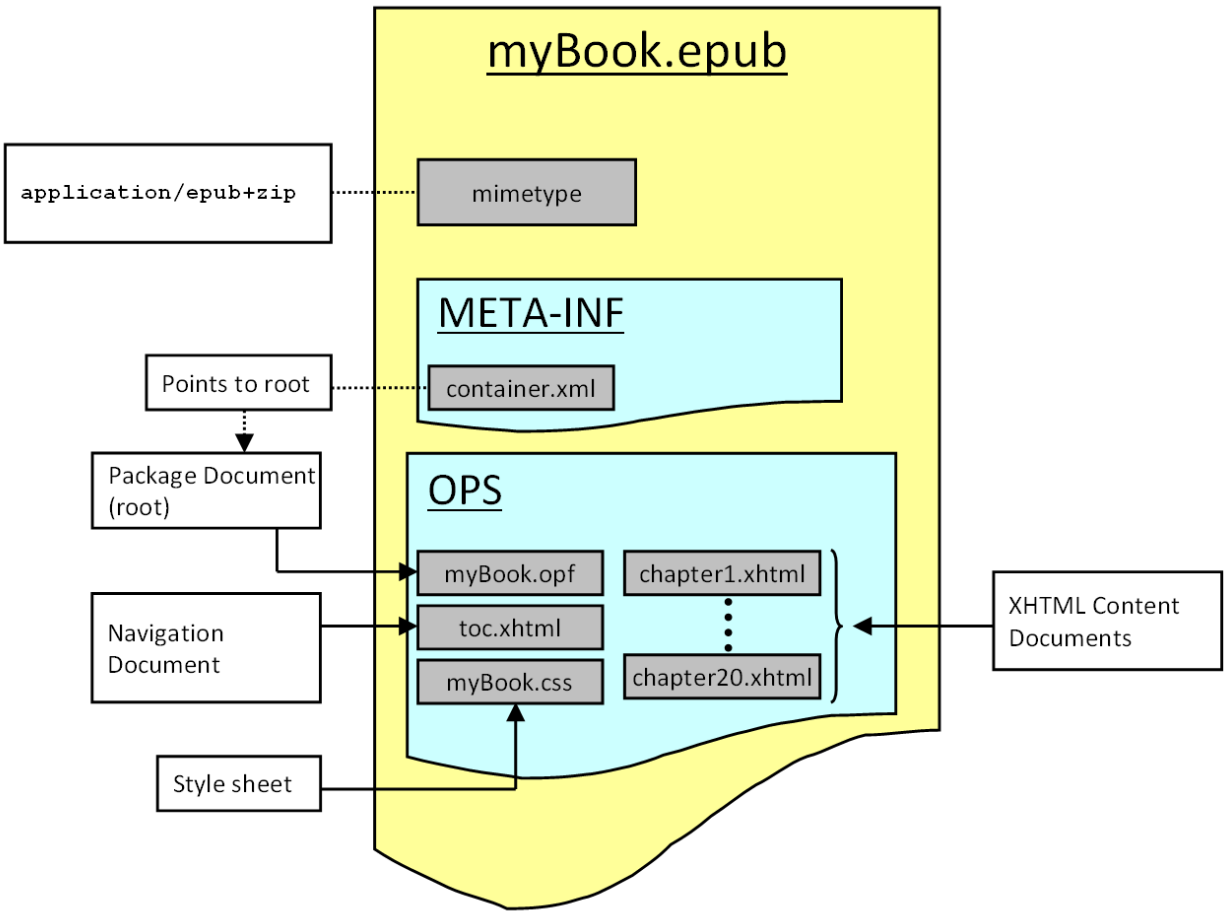

Figure 1 Structure of an EPUB 3 file. Gray rectangles are files, all other elements represent directories.

---

[4] In *EPUB* 2 this is called "Open Container Format" (*OCF*).

[5] As this was introduced in *EPUB* 3, there is no *EPUB* 2 equivalent. Also, note that *EPUB Media Overlays* are outside the scope of the current document.

As explained in the previous section, it is an ordinary *ZIP* archive. Inside it is a file system. At the highest level there are three elements:

1. A *mimetype* file, which must be uncompressed. It contains a text string that enables viewer applications to identify the file as *EPUB*.

2. A directory called *META-INF*. It contains a *container.xml* file, which points to one or more *root* files. A *root* file is the *Package Document*, which is described below [6].

3. A directory called *OPS*. It contains all *Content Documents*, including:

    - A *Package Document*, which holds bibliographic and structural metadata. Importantly, it specifies all the publication's Publication Resources, including their locations.

    - A *Navigation Document* (Table of Contents).

    - A set of *XHTML Content Documents* (in the example each corresponds to one book chapter).

    - A style sheet.

2.7 Note on file and directory naming

The naming of files and directories, and the structure of their contents, are not fixed. Exceptions are the *mimetype* file, and the *META-INF* directory (and its contents). These files and this directory are compulsory, and they should be located at these exact respective locations within each *EPUB* file. The document contents (which are stored in an *OPS* directory in this example) may be organised in any arbitrary manner, provided that the organisation is defined in the *Package Document* (which is located through *container.xml* in *META-INF*, and this file should *always* be present at this exact location).

The example in Figure 1 is fairly minimal, as an *EPUB* publication may also contain images, multimedia files, fonts, scripts and additional metadata. Also, text content of *EPUB* 2 publications may be in *DTBook* [10] format rather than *XHTML*. This will all be covered in more detail in the following chapters.

In the following chapters we will examine these standards in more detail.

[6]The META-INF directory may also contain a number of optional files, which are related to encryption, digital rights management, and metadata. In addition it may contain a file with digital signatures of individual resources (i.e. files), which is useful for establishing the authenticity of the publication's contents

3 Core media types

The previous chapter explained how an *EPUB* file is actually a *ZIP* archive that contains a collection of separate file objects, which are called *Publication Resources*. These Publication Resources can have a number of file formats. The allowed formats are defined in the *EPUB Publications* specification as a list of *Core Media Types*. A Core Media Type is defined as "a set of Publication Resource types for which no fallback is required" [6].

Put simply, the Core Media Types define a set of file formats that *must* be supported by any *EPUB* viewer application. Resources that have a different format are called *Foreign Resources*. Foreign Resources are allowed in *EPUB*, but there are some restrictions. Most importantly, there must be a fallback to a Core Media Type in case the Foreign Resource cannot be rendered. An example: suppose that an *EPUB* file contains a *JP2* (JPEG 2000 Part 1) image. *JP2* is not a Core Media Type, and viewer applications may not be able to render such resources. In this case, the *EPUB* file must contain an alternative representation of the image, using a format that *is* a Core Media Type. This could be, for example, a JPEG image.

3.1 Core media types in *EPUB* 2 and 3

Tables 1 and 2 below list the Core Media Types of *EPUB* 2 and *EPUB* 3, respectively.

Media Type	Description
image/gif	GIF Images
image/jpeg	JPEG Images
image/png	PNG Images
image/svg+xml	SVG documents
application/xhtml+xml	XHTML1.1 Content Documents
application/x-dtbook+xml	Digital Talking Book Documents
text/css	CSS 2.0
application/xml	XML
text/x-oeb1-document	Open eBook Publication Structure Document (deprecated)
text/x-oeb1-css	Open eBook Publication Structure CSS (deprecated)
application/x-dtbncx+xml	NCX

Table 1 Core Media Types EPUB 2.

Media Type	Description
image/gif	GIF Images
image/jpeg	JPEG Images
image/png	PNG Images
image/svg+xml	SVG documents
application/xhtml+xml	XHTML5 Content Documents, EPUB Navigation Document
application/x-dtbncx+xml	NCX (superseded)
application/vnd.ms-opentype	OpenType fonts
application/font-woff	WOFF fonts
application/smil+xml	EPUB Media Overlay documents
application/pls+xml	Text-to-Speech Pronunciation lexicons
audio/mpeg	MP3 audio
audio/mp4	AAC LC audio using MP4 container
text/css	EPUB Style Sheets
text/javascript	Scripts

Table 2 Core Media Types EPUB 3.

Note that *EPUB* 2 contains two deprecated Core Media Types, both of which are remnants from the *Open eBook* format that was the predecessor of *EPUB*. Most of the Core Media Types of *EPUB 3* are also supported by *EPUB* 2, with the exception of fonts (*OpenType* and *WOFF*), Media Overlay Documents, Text-to-Speech Overlay documents, *MP3* and *AAC LC* Audio and scripts.

In addition, for Content Documents *EPUB* 2 allowed the use of the *Digital Talking Book* format [10] as an alternative to *XHTML1.1*. This is no longer the case in *EPUB* 3. However, Section 2.2 (Reading System Conformance) of *EPUB Publications* (which is part of *EPUB* 3) states that "[a]n EPUB Reading System (...) *should* process EPUB version 2 Publications (...)"[6]. This means that most viewer applications should be able to render *EPUB* 2 publications without problems [7].

Finally, although both *EPUB* versions may (and usually will) contain *XHTML*, *EPUB* 2 uses *XHTML1.1*, whereas *EPUB* 3 is based on *XHTML5*. The main implication of this is that *EPUB* 3 files may not render correctly on viewers that were designed for *EPUB* 2.

[7]Besides, the *DTBook* format is also *XML*-based, which makes the interpretation of these files relatively straightforward

4 Package and Navigation Document

4.1 The Package Document

The *Package Document* is a resource that contains bibliographic and structural metadata. It is the primary source of information about an *EPUB* publication and its resources. Figure 2 shows a minimal example.

In most cases, The Package Document is made up of the following 3 elements:

- A *metadata* element, which contains (mainly bibliographic) metadata.
- A *manifest* element (known as the *Publication Manifest*). This identifies and describes the resources that make up a publication. It includes the *MIME* type [3] of each resource.
- A *spine* element, which specifies the default reading order.

The above elements are required, and are present in any *EPUB* publication. In addition, two optional elements may exist that are not discussed here. Further details can be found in Section 3.1 of *EPUB Publications 3.0* [6].

4.2 The Navigation Document

The function of the *Navigation Document* is to provide a mechanism to navigate a publication. It is essentially a hierarchical table of contents. In *EPUB* 2, the Navigation Document is an *.ncx* resource. *NCX* is an acronym of *Navigation Control file for XML applications*. It is an *XML* resource, and its precise makeup is defined by [11]. In *EPUB* 3 the *.ncx* file is superseded by a *Navigation Document* in *XHTML* format [7]. An *EPUB* 3 publication may nevertheless contain an *NCX* resource for compatibility reasons. This will enable *EPUB* 2 viewer applications to read *EPUB* 3 publications [8].

The navigation document also supports the definition of page numbers. This is done through the *pageList* element in the *.ncx* file (*EPUB* 2), or the *page-list nav* element in *EPUB* 3 [7].

[8] If such publications contain features that are not defined by the *EPUB* 2 format, they may not be displayed properly.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <package xmlns="http://www.idpf.org/2007/opf" version="3.0" xml:lang="en"
    unique-identifier="pub-id" prefix="cc: http://creativecommons.org/ns#">
  - <metadata xmlns:dc="http://purl.org/dc/elements/1.1/">
      <dc:title id="title">Moby-Dick</dc:title>
      <meta refines="#title" property="title-type">main</meta>
      <dc:creator id="creator">Herman Melville</dc:creator>
      <meta refines="#creator" property="file-as">MELVILLE, HERMAN</meta>
      <meta refines="#creator" property="role"
        scheme="marc:relators">aut</meta>
      <dc:identifier id="pub-id">code.google.com.epub-samples.moby-dick-
        basic</dc:identifier>
      <dc:language>en-US</dc:language>
      <meta property="dcterms:modified">2012-01-18T12:47:00Z</meta>
      <dc:publisher>Harper & Brothers, Publishers</dc:publisher>
      <dc:contributor id="contrib1">Dave Cramer</dc:contributor>
      <meta refines="#contrib1" property="role"
        scheme="marc:relators">mrk</meta>
      <dc:rights>This work is shared with the public using the Attribution-
        ShareAlike 3.0 Unported (CC BY-SA 3.0) license.</dc:rights>
      <link rel="cc:license" href="http://creativecommons.org/licenses/by-
        sa/3.0/" />
      <meta property="cc:attributionURL">http://code.google.com/p/epub-
        samples/</meta>
    </metadata>
  - <manifest>
      <item id="font.stix.regular" href="fonts/STIXGeneral.otf" media-
        type="application/vnd.ms-opentype" />
      <item id="toc" properties="nav" href="toc.xhtml" media-
        type="application/xhtml+xml" />
      <item id="titlepage" href="titlepage.xhtml" media-
        type="application/xhtml+xml" />
      <item id="cover" href="cover.xhtml" media-
        type="application/xhtml+xml" />
      <item id="cover-image" properties="cover-image"
        href="images/9780316000000.jpg" media-type="image/jpeg" />
      <item id="style" href="css/stylesheet.css" media-type="text/css" />
      <item id="xchapter_001" href="chapter_001.xhtml" media-
        type="application/xhtml+xml" />
      <item id="xchapter_002" href="chapter_002.xhtml" media-
        type="application/xhtml+xml" />
      <item id="xchapter_003" href="chapter_003.xhtml" media-
        type="application/xhtml+xml" />
      <item id="brief-toc" href="toc-short.xhtml" media-
        type="application/xhtml+xml" />
    </manifest>
  - <spine>
      <itemref idref="cover" linear="no" />
      <itemref idref="titlepage" linear="yes" />
      <itemref idref="brief-toc" linear="yes" />
      <itemref linear="yes" idref="xpreface_001" />
      <itemref linear="yes" idref="xchapter_001" />
      <itemref linear="yes" idref="xchapter_002" />
      <itemref linear="yes" idref="xchapter_003" />
      <itemref idref="toc" linear="no" />
    </spine>
  </package>
```

Figure 2 Minimal Package Document.

5 Functionality

5.1 Layout and appearance capabilities

The layout and appearance capabilities of *EPUB* are not directly relevant for preservation. However, they do put constraints on which materials are suitable for representing in the format. Since *EPUB* 2 and 3 show significant differences in this area, this section provides a brief discussion with some examples.

Reflowable content versus fixed page layout

Before getting into any details about *EPUB*'s layout capabilities, it is important to stress that *EPUB* is primarily a *reflowable* document format: content (e.g. text) is presented in a way that fits the viewer device, the viewer software, or the user's preferences. For instance, changing the text size, or re-sizing the viewer window will cause text to dynamically re-flow to fit the viewer. The Figures 3, 4, 5 and 6 illustrate this (for this example I used Adobe's Digital Editions viewer). This behaviour is unlike *PDF*, which uses fixed-size pages, where all page elements (text, images) are positioned at a fixed position on each page, at a fixed size. However, fixed-page layouts are possible in *EPUB* 3 (see discussion below).

Figure 3 Page 9 of 'Lanseloet van Denemarken', note 2-column view

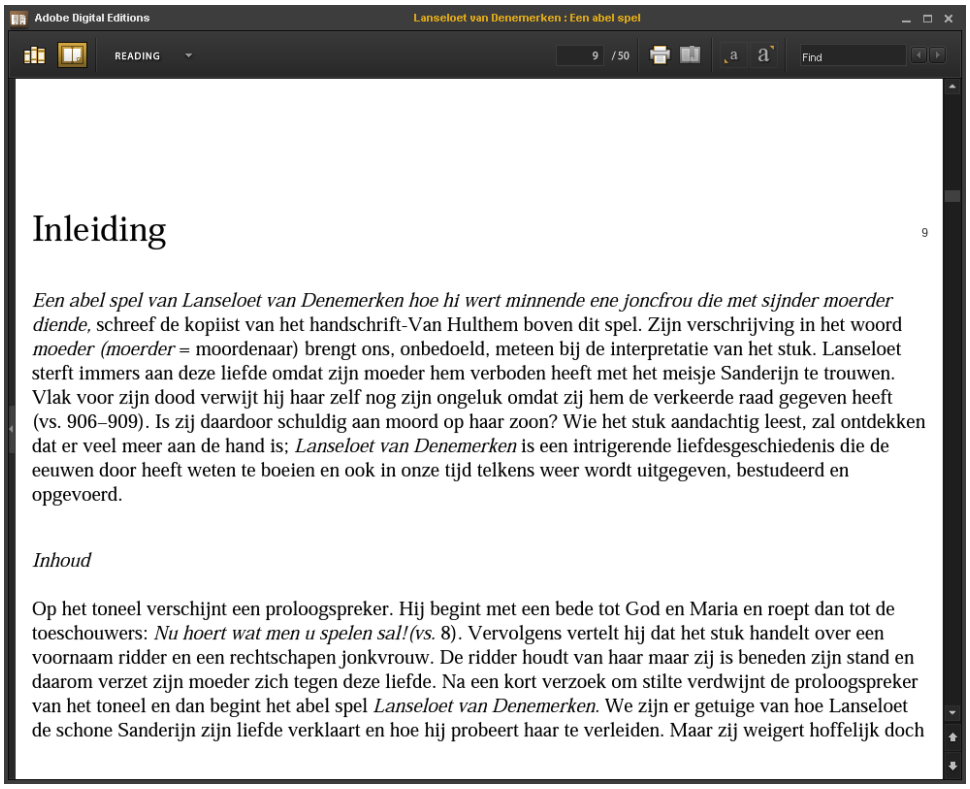

Figure 4 Page 9 of 'Lanseloet van Denemarken' after window resize, note how text reflows to 1-column view.

Figure 5 Page 9 of 'Lanseloet van Denemarken' after increasing text size.

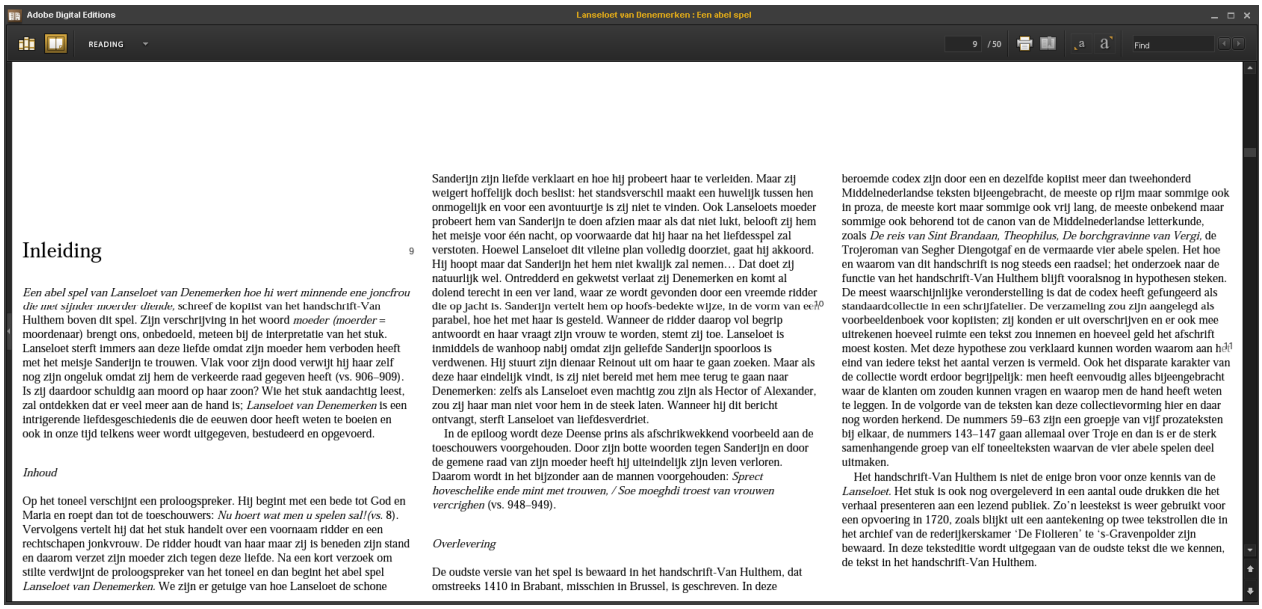

Figure 6 Page 9-11 of 'Lanseloet van Denemarken' after decreasing text size. Note how text reflows to 3-column view.

EPUB 2

EPUB 2 only provides limited functionality for controlling the appearance and layout of a publication. As noted by Kasdorf [12], the *International Digital Publishing Forum* has its origins in the trade publishing sector, and the earlier versions of *EPUB* reflected this. This means that the format is adequate for most "general audience books" (such as fiction). It is less suitable for textbooks, scientific and technical publications, newspapers, comic books, and so on. The main reason is that such publications often require more advanced layout and typographic capabilities, and these are lacking in *EPUB* 2.

EPUB 3

One of the aims of *EPUB* 3 was to overcome the layout and typography limitations of *EPUB* 2, and through the use of *XHTML5* more sophisticated layouts are possible. *EPUB* 3 also supports a subset of the *MathML* markup language, which is used to represent mathematical equations [9]. Interestingly, the *EPUB* 3 specification acknowledges that reflowable content may not always be desired, and that in specific cases "content and design are so intertwined they cannot be separated". For such situations, it offers the possibility to create *fixed-layout* documents[13]. As viewer support for *EPUB* 3 is still virtually non-existent, it was not possible to do any further assessment of *EPUB* 3's advanced layout capabilities at this stage.

[9]In *EPUB* 2 mathematical equations can only be represented using graphics files (e.g. *PNG*)

Significance for preservation

The importance of the above appearance and layout features is that they largely determine which content can be suitably represented in *EPUB* format. The scope of *EPUB* 2 is limited to content that doesn't require any advanced layout and typography. *EPUB* 3 is designed to support a much wider range of publications.

5.2 Multimedia

Starting with version 3, *EPUB* supports publications that contain audio and video. *EPUB* 2 has no multimedia support.

Significance for preservation

Publications that contain multimedia are more difficult (and costly) to preserve, as they generally have more technical dependencies than simple text and image content.

5.3 Scripting

Another addition to *EPUB* 3 is the possibility to include *Javascript* resources. This makes it possible to add interactivity.

Significance for preservation

Support for scripting introduces the possibility to include malicious code, which is a security risk. It also raises concerns about privacy, since the scripting functionality can be used to track information about consumers [14].

6 Processing EPUB: characterisation

Ingest and pre-ingest workflows usually include one or more processing steps to obtain basic information about a digital object, which is needed for its preservation. This is called *characterisation*, which can be sub-divided into:

- **Identification**: establishes the object's format.
- **Validation**: verifies whether the object conforms to the format'specification.
- **Feature extraction**: extracts (mostly technical) meta-information about the object.

Therefore, it is important that software tools exist that provide sufficient support for *EPUB*. This chapter gives an overview of some (potentially) useful tools for identification, validation and feature extraction. Also included here are a number of preliminary tests. It is important to stress here that the aim of this chapter is primarily to give a first impression of tools that may be relevant to *EPUB*. It is *not* intended to be exhaustive, and an in-depth evaluation of these tools could be the subject of a follow-up to this report.

6.1 Test data

For all tests in this chapter the following data sets were used:

- **PUB_A**: a set of 91 *EPUB* 2 files by publisher ■■■■■■■
- **PUB_B**: a set of 5 *EPUB* 2 files by publisher ■■■■■■■
- **EPUB3**: a set of 26 *EPUB* 3 files by the *International Digital Publishing Forum* [15]

6.2 Identification

DROID

DROID (Digital Record Object Identification) is a tool that identifies digital objects using PRONOM format signatures ('magic numbers') and/or known file extensions. The identification results are reported as *PRONOM*-compliant Persistent Unique Identifiers (*PUIDs*). *DROID* is an open-source, platform-independent *Java* application. It can be used directly from the command line, or, alternatively, using a graphical user interface [16]. Here *DROID* 6.01 was used, with signature file V. 59.

Fido

Fido (Format Identification for Digital Objects) is an identification tool that also uses the *PRONOM* format signatures. It is essentially a *DROID* clone [17]. Here version 1.0.0 was used.

Unix File Utility

File is a command-line utility that is part of every major Unix and Unix-like operating system. It identifies files based on signatures ('magic numbers') stored in a 'magic' file. The

file utility reports identification results as *MIME* types (and optionally as a text description) [18]. Here *File* 5.11 was used on a Windows-based system using the *Cygwin* environment[10].

Apache Tika

Tika is a software toolkit that "detects and extracts metadata and structured text content from various documents using existing parser libraries" [31]. Like *File*, it identifies files based on signatures ('magic numbers'), and the results are reported as *MIME* types. Here *Tika* 1.2 was used.

Test results

I ran *DROID*, *Fido*, *File* and *Tika* on the test data. The tables below show the results.

Data	ID result
PUB_A	x-fmt/263,application/zip (91)
PUB_B	x-fmt/263,application/zip (5)
EPUB3	x-fmt/263,application/zip (26)

Table 3 Identification results, *DROID* + *Fido* (results as PUID / Internet Media Type).

Data	ID result
PUB_A	application/epub+zip (91)
PUB_B	application/zip (5)
EPUB3	application/epub+zip (26)

Table 4 Identification results, *File* (results as Internet Media Type).

Data	ID result
PUB_A	application/epub+zip (91)
PUB_B	application/epub+zip (5)
EPUB3	application/epub+zip (26)

Table 5 Identification results, *Tika* (results as Internet Media Type).

Both *DROID* and *Fido* identify all *EPUB*s as simple *ZIP* files. *File* correctly identifies all files in the *PUB_A* and *EPUB3* data sets. The *PUB_B* files are identified as simple *ZIP* files. This is not related to any shortcoming of *File*, but caused by the fact that the *PUB_B* files are no valid *EPUB*s (see next section). *Tika* identifies *all* files as *EPUB*, including the (malformed) *PUB_B* ones[11].

The implication of the above is that of the tested tools both *File* and *Tika* are able to automatically establish the file format of an *EPUB* publication. For *DROID* and *FIDO* the obvious solution would be to create *EPUB* file signatures.

[10]Link: http://cygwin.com/

[11]As already pointed out before, no official Internet Media Type exists for *EPUB* at this stage, although the unofficial *application/epub+zip* is often used.

6.3 Validation: *EpubCheck*

EpubCheck is a tool that validates *EPUB* files, and it can be used to detect various errors. Most of its development is done by Adobe Systems Incorporated (which is also the copyright owner). It is an open-source, platform-independent *Java* application. It has a command line interface, and can be used as a *Java* library as well. The Wiki on the tool's homepage [19] includes a detailed description of the validation procedure. For the tests here *EpubCheck* 3.0b5 was used, which supports both versions 2 and 3 of the *EPUB* format.

Overview of results

I used *EpubCheck* to validate the *EPUB* files in each test dataset. The table below summarises the results:

Data	Warnings	Errors
PUB_A	1 (1 %)	2 (2 %)
PUB_B	0 (0 %)	5 (100 %)
EPUB3	4 (7 %)	2 (15 %)

Table 6 Number of files with warnings and errors, *EpubCheck*.

What is particularly noteworthy is that *all* files in the *PUB_B* data set resulted in errors. In the following sections we will look at these results in more detail.

Results *PUB_A*

Two files in the *PUB_A* data set resulted in errors:

- For one file, the Package Document (*OPF*) contains an incorrectly formatted date.
- For another one, it reported that the "id" attribute -which is part of the *navPoint* element in the Navigation Document (*NCX*)- does not have a unique value.

For one file *EpubCheck* gave a warning that the spine (part of the Package Document) "contains only non-linear resources".

Results *PUB_B*

None of the files in the *PUB_B* data set passed validation by *EpubCheck*. They all produced the following errors:

- *Mimetype contains wrong type (application/epub+zip expected)* – the Open Container Format specification states that the *mimetype* resource must be the *first* file in the container, and that it must be *uncompressed*. A check in a hex editor revealed that the *mimetype* resource in the *PUB_B* files is actually *compressed*. As a result, the correct media type (which should be the text string *application/epub+zip*, starting at byte offset 38) cannot be established. This also explains why *File* cannot correctly identify the *PUB_B EPUB*s. Interestingly, Apache *Tika* is nevertheless able to identify them.
- *File listed in reference element in guide was not declared in OPF manifest* - this is caused by the fact that the Package Document (*OPF*) contains references to resources that are not declared in the *Publication Manifest*.

- *Referenced resource missing in the package* - essentially the same as above: the Package Document (*OPF*) contains references to resources that are not part of the package.

In addition, for one file the following error was reported as well:

- *The "id" attribute does not have a unique value* (see above)

Results *EPUB3*

The analysis of the *EPUB3* data set yielded the following error for 2 files:

- *Font-face reference OPS/Font/AGaramondPro-Regular-0908.otfto non-standard font type application/x-font-otf* - appears to be caused by a reference to a *mimetype* (*application/x-font-otf*) that is not a Core Media Type.

In addition, the following warning messages occurred:

- *resource EPUB/OldStandard-Regular.obf.otf cannot be decrypted* - this message occurred for a file with embedded fonts that use font obfusciation.
- *item (EPUB/examples/figure-gallery-example.html) exists in the zip file, but is not declared in the OPF file*
- *File name contains non-ascii characters* - this warning message was produced for a Japanese-language file that contains resources that have names which include non-ascii characters.

Deployment in automated workflows

One potential problem of *EpubCheck* is that it produces output that is fairly unstructured. If a file passes all tests, the message "No errors or warnings detected" is written to the standard *output* device (*stdout*). If the validation results in any errors or warnings, these are written to the standard *error* device (*stderr*). However, *stderror* is used for reporting internal *EpubCheck* exceptions as well, and *stdout* is also used for informational messages. Having said this, deployment of *EpubCheck* in an automated workflow is certainly possible, but it would require some tweaking of its output.

6.4 Validation: *FlightCrew*

FlightCrew is another *EPUB* validator that was developed as an alternative to *EpubCheck*. Its authors claim that it is a 'better' *EPUB* validator than *EpubCheck*. *FlightCrew* is written in C++, and released under an open-source license. It comprises a library, a command-line interface and a graphical user interface [32]. All tests were done using *FlightCrew* 0.7.2, which only supports version 2 of the *EPUB* format.

Overview of results

I used *FlightCrew* to validate all *EPUB* 2 files in the test data (i.e. *PUB_A* and *PUB_B*; *EPUB3* isn't included as *FlightCrew* doesn't support *EPUB* 3). The table below summarises the results:

Data	Warnings	Errors
PUB_A	5 (5 %)	9 (10 %)
PUB_B	0 (0 %)	5 (100 %)

Table 7 Number of files with warnings and errors, *Flightcrew*.

Again *all* files in the *PUB_B* data set resulted in errors. In the following sections we will look at these results in more detail.

Results *PUB_A*

In the *PUB_A* data set the following errors were reported:

- For two files, the Package Document (*OPF*) contains an incorrectly formatted date.
- For three files, the occurrence of a non-allowed elements was reported in *HTML* resources (‘*element 'a' is not allowed for content model '(br|span|bdo|map|object|img|svg|tt|i|b|big|small|em|strong|dfn|code|q|samp|kbd|var|cite|abbr|acronym|sub|sup|input|select|textarea|label|button|ins|del|script)"*)
- For four files, errors were reported on items in the Package Document (*OPF*) that are using the wrong media type. In most cases these errors referred to embedded OpenType fonts (‘*The <item> element's "media-type" attribute has value "application/vnd.ms-opentype", but the file's media type is "application/x-font-ttf".*’). This appears to be related to the fact that there is no ‘official’ *mimetype* for OpenType fonts. *FlightCrew* enforces the use of *application/x-font-ttf* (claiming that this is the *de facto* standard), whereas the *EPUB* specifications use *application/vnd.ms-opentype* throughout [33].

In addition, the following warning was reported for 6 files:

- ‘*This resource is present in the OPF <manifest>, but it's not reachable (it's unused).*’ In each case it refers to embedded font files that are declared in the manifest, but which are not actually used (i.e. referred to elsewhere).

Results *PUB_B*

Similar to *EpubCheck*, none of the files in the *PUB_B* data set passed validation by *FlightCrew*. They all produced the following errors:

- *Bytes 30-60 of your epub file are invalid* – this is caused by the fact that the *mimetype* resource in the *PUB_B* files is not uncompressed.
- *This resource is reachable but not present in the OPF <manifest>* - this is caused by the fact that the Package Document (*OPF*) contains references to resources that are not declared in the *Publication Manifest*.
- *This OPS document is reachable but not present in the OPF <spine>* - essentially the same as above: the Package Document (*OPF*) contains references to resources that are not part of the package.
- *The <reference> element's "type" attribute has value "copyright" which is not an OPF-specified value*

In addition, for three files the following error was reported as well:

- *The <item> element's "media-type" attribute has value "application/vnd.adobe.page-template+xml", but the file's media type is "application/vnd.adobe-page-template+xml* – again this is caused by a supposedly incorrect *mimetype* declaration.

Deployment in automated workflows

Much like *EpubCheck*, *FlightCrew*'s output is pretty unstructured and not optimally suited to automatic processing. . If a file passes all tests, the message "No problems found" is written to the standard *output* device (*stdout*). If the validation results in any errors or warnings, these are written to the standard *error* device (*stderr*). Having said this, deployment of *FlightCrew* in an automated workflow is certainly possible, but it would require some tweaking of its output. Also, unlike *EpubCheck*, *FlightCrew* has a unique number assigned to each specific error, which may also make automated processing somewhat easier.

6.5 Validation: *EpubCheck vs FlightCrew*

It is beyond the scope of this report to provide a detailed comparison between *EpubCheck* and *FlightCrew*. However, from the tests some preliminary observations can be made. First of all, *FlightCrew* generally appears to be more strict than *EpubCheck*, resulting in more reported errors. In some cases the nature of these "errors" is debatable (this applies in particular to the "media-type" related errors, since the affected media types are often not well defined). For the tested files both tools were generally in agreement (although *FlightCrew* reported more errors). Interestingly, whereas *EpubCheck* identified a non-unique "id" attribute in the Navigation Document (*NCX*) in one of the *PUB_A* files, *FlightCrew* didn't report any similar error. Unfortunately *FlightCrew*'s documentation is pretty minimal compared to *EpubCheck*, which makes it difficult to make a comparison between both tools.

Also, the error messages generated by *FlightCrew* are generally more verbose and specific than those reported by *EpubCheck*. Unlike *EpubCheck*, each error in *FlightCrew* has a unique number, which may make automated processing somewhat easier. A limitation of *FlightCrew* is that it doesn't (yet) support *EPUB* 3. Also, even though a formal assessment of the computational performance was not part of the current evaluation, *FlightCrew* appeared to be comparatively slow, on one occasion needing almost 3 minutes (!) to analyse a single 1.3 MB file. So, concluding, more elaborate testing would be needed to decide on either validation tool in any operational setting.

6.6 Feature extraction

Epub-tools

Epub-tools is an open-source "suite of command-line utilities for creating and manipulating epub book files" [20]. It is written in *Haskell*, and the author is Dino Morelli. It includes an *epubmeta* utility, which extracts and displays metadata from an *EPUB* file. However, it only extracts information from the Package Document. It does not provide any information on whether an *EPUB* contains encrypted content or uses digital rights management technology. It is unclear to what extent *EPUB* 3 is supported (although some quick tests on the *EPUB3* data set showed that *epubmeta* is able to extract metadata from these files).

All in all, *epub-tools* wil most likely not be overly useful in a pre-ingest workflow. It doesn't provide any information about those very features of *EPUB* that, within an archiving context, are potentially the most problematic of the format.

Apache Tika

Tika [31], which was already mentioned in section 6.2, also includes metadata extraction functionality. However, like *Epub-tools*, it o extracts metadata from the Package Document, without providing any information on encryption or digital rights management technology.

Other feature extraction tools

To the best of my knowledge there are currently no other feature extraction tools for *EPUB*. However, since the format is based on established standards (e.g. *ZIP*, *XML*) that are widely supported by existing programming libraries, building such a tool would be fairly straightforward. It is also something that could be done with limited amounts of time and resources.

7 EPUB as a preservation format

This chapter discusses the suitability of *EPUB* as a preservation format. The criteria used here are taken (and in a number of cases adapted) from [21] (The National Archives) and [22] (Library of Congress). If relevant, the outcome of each of these criteria is discussed separately for *EPUB* 2 and *EPUB* 3. Readers should also be aware that the *Sustainability of Digital Formats* website of Library of Congress contains entries on both *EPUB* 2 and 3 [23],[24].

7.1 Ubiquity, support and interoperability

EPUB 2

The *Wikipedia* entry *Comparison of e-book formats* states that "as of 2011 (*EPUB*) is the most widely supported vendor-independent XML-based e-book format" [25]. The entry on *EPUB* lists 22 different viewers that are supporting the format [4], including several open source implementations. Popular ones are, for example, Adobe Digital Editions [12] and the open-source Calibre software[13]. An elaborate discussion of the pros and cons of specific viewers is beyond the scope of this report, but a useful review of some of the most widely-known ones can be found in [26]. *EPUB* 2 is also supported by many hardware-based reading systems. Even though *EPUB* 2 is the most popular *vendor-independent* e-book format, the current e-book market is dominated by proprietary formats, making *EPUB* a relatively minor player overall.

EPUB 3

At this moment, there are (to my knowledge) no readers that support *EPUB* 3. The review in [26] lists one reader (out of a total of 21) as supporting *EPUB* 3; however, it also states that it doesn't (yet) render these files correctly. I did some preliminary tests on the support of multimedia files in two popular *EPUB* readers: *Adobe Digital Editions* (version 1.7.2) and *Calibre* (0.8.49). Both readers were unable to render *MP4* video and *MP3* audio in an *EPUB* 3 file that is part of a set of sample data from *IDPF* [15]. Nevertheless, these files could still be opened, and the text content was readable. An *EPUB* 3 *Fixed Layout* document could also be opened in both *Digital Editions* and *Calibre*, but both viewers failed to render the page layout correctly. *Digital Editions* only displayed part of each page, whereas *Calibre* ended up inserting blank pages. Both viewers were also unable to correctly render a document containing *MathML* equations. Since the *EPUB* 3 specification was only released in October 2011, support of this version is likely to improve in the future. The International Digital Publishing Forum (which is also the organisation behind *EPUB*) has initiated work on an open-source reference implementation called *Readium*. This is a set of libraries for viewing and creating *EPUB* content. An early beta version of the software is available as an extension to *Google*'s *Chrome* web browser [27].

---

[12]Link: http://www.adobe.com/products/digitaleditions/

[13]Link: http://calibre-ebook.com/

Significance for preservation

Popular formats that are widely supported by (open source) viewer software are unlikely to become inaccessible over time. Formats without sufficient use and support are at risk of becoming obsolete, and long-term access may become problematic.

7.2 Disclosure

EPUB is an open format. The format's specifications are freely available from the website of the *International Digital Publishing Forum* [2].

Significance for preservation

For preservation open formats are preferable to closed ones (which are typically proprietary formats whose specifications are trade secrets). Openness ensures that all technical information about the format is publicly available, and avoids dependence on specific software by the owner of the format (vendor lock-in).

7.3 Documentation quality

The documentation of both *EPUB* 2 and 3 is both comprehensive and complete.

Significance for preservation

Documentation should be of sufficient quality to allow interpretation of files that use the format. This includes the development of new access (e.g. viewer) software.

7.4 Stability

File formats that are suitable for long-term preservation have format specifications that are stable over time, without being subject to constant or major changes between versions [21]. However, a comparison between the specifications of *EPUB* 2.0 and *EPUB* 3 shows a number of major changes. Most notable are:

- Deprecation of *DTBook* content documents
- Change from *XHTML1.1* to *XHTML5*
- Change from *CSS2* to a subset of *CSS3*
- Deprecation of *NCX* navigation documents
- Added support for audio video and scripts

Also, it is remarkable (and a little confusing) that even the *names* of the standards that make up *EPUB* were completely changed between *EPUB* 2 and 3.

Moreover, *EPUB* 3 is heavily dependent on *(X)HTML5* and *CSS3*, standards that are still unfinished "works in progress". A formal Recommendation document for *HTML5* is currently expected to be finalised some time in 2014 [28]. In the meantime, *EPUB* 3 will be based on an unfinished standard. This is acknowledged by Kasdorf, who defends the decision to use *XHTML5* and *CSS3* by stating that the *EPUB* 3 Working Group selectively specified "those modules of *HTML5* and *CSS3* that either (1) are in fact considered finished,

for all practical purposes, or (2) are essential to an *EPUB* 3 requirement and are close enough to resolution that they are *reasonably safe to use*" (emphasis added by JvdK). He adds that *if HTML5* or *CSS3* would "change from what *EPUB* 3 is specifying, *EPUB* 3 makes the commitment to change along with it". He also calls this approach "realistic, practical and *not as risky as it appears to be*" [12]. He does not further substantiate this claim (apart from one general remark about *EPUB* 3's use of *CSS*).

Summarising, various basic features from *EPUB* 2 have been deprecated or replaced in *EPUB* 3, and the latter's dependence on unfinished standards (especially *HTML5*) implies that further (probably more minor) changes are likely. This lack of stability of the *EPUB* specification does raise some concerns about the suitability of *EPUB* as a format for long-term preservation. The assessment by Library of Congress expresses similar concerns [23],[24].

Significance for preservation

If a format is subject to constant major changes over time, this is likely to have a detrimental effect on overall interoperability. Specific versions of the format may only be rendered correctly on specific viewers, and providing backward compatibility will become more difficult.

7.5 Ease of identification, validation and feature extraction

The tests in the previous chapter revealed the following:

- Out of the tested identification tools (Unix *File*, *DROID* and *Fido*), both *File* and *Tika* are able to correctly identify *EPUB* files. *DROID* and *Fido* only identify EPUB to the level of the container format (i.e. *ZIP*). On a related note, *EPUB* is not yet included in *PRONOM*, and only an unofficial Internet Media Type exists at this time.

- Validation is possible using *Epubcheck* and *FlightCrew*. Both tools were capable of detecting a variety of errors in the test data sets (although FlightCrew doesn't support *EPUB* 3 yet). However, not all errors detected by *FlightCrew* are also picked up by *Epubcheck*, and vice versa. Also, *FlightCrew* is finicky (i.e. will report errors) for *mimetype* declarations that are literally given in the *EPUB* specification. More elaborate testing would be necessary to assess both tools' true usefulness. If any of them are used as part of an automated workflow, one should be aware of the somewhat peculiar way they are reporting their output.

- Feature extraction is still rather problematic. Both the metadata-extraction utility from *epub-tools* and *Tika* only extract information from an *EPUB*'s Package Document. This is not sufficient for detecting files that may contain technical protection technology (encryption).

Significance for preservation

The availability of identification, validation and feature extraction tools ensures that file objects can be adequately processed, necessary technical (preservation) metadata can be extracted, and basic quality checks are possible.

7.6 Intellectual Property Rights

Neither *EPUB* nor any of the standards and formats that are associated with *EPUB*'s core media types are subject to any known patents or licensing restrictions.

Significance for preservation

Formats may use technology that is subject to patents or other intellectual property restrictions. As these may limit the (future) use of files of such formats, for preservation formats are preferred that not subject to intellectual property restrictions.

7.7 Technical protection

EPUB publications (both 2 and 3) may use Digital Rights Management (DRM) technology. The use of DRM usually means that (part of) the contents of the publication are encrypted. Depending on the exact implementation, viewing such files may be restricted to a registration key that is coupled to specific software, or it may require a password. The *EPUB* 3 OCF specification does not prescribe any specific DRM method, but it says that this may change in future versions [8]. The current situation is that various vendor-specific methods exist, and *any* method is allowed. For instance, both Sony and Adobe are using their own DRM methods [29].

EPUB also supports encryption as a means to prevent the re-use of embedded fonts. Because of the use of *ZIP* as a container format, it is relatively easy to extract and re-use fonts that are embedded within an *EPUB* file. This raises a problem for most commercially available (i.e. non-free) fonts. In an attempt to discourage users from re-using embedded fonts, the *EPUB* specification includes a technique called *font obfusciation*. It works by encrypting the first 1040 bytes of a font file. The encryption method used is relatively simple, and the specification states that a determined user will be able to gain access to the unencrypted font. As such, it is simply meant as a "stumbling block for those who are unaware of the license details of the supplied font" [8].

The first type of encryption (DRM-related) poses a direct threat to the accessibility of *EPUB* publications, and publications that use this technology should not be accepted for ingest. The second type (*font obfusciation*) only affects embedded fonts. Since *EPUB* viewers are required to handle obfusciated fonts,[14] accessibility is not immediately at risk. It *may* affect future migrations to some other format. If the migration tool cannot handle obfusciated fonts, this will alter the appearance of the migrated object. However, because of the simple nature of the encryption involved, decrypting the font files is pretty straightforward. Legal constraints may nevertheless prohibit doing so.

EPUB publications that contain DRM can be identified by the presence of a *rights.xml* resource in the *META-INF* directory. An *encryption.xml* resource in the same directory indicates the presence of encrypted content.

[14]This only applies to viewers for *EPUB* 3; *EPUB* 2 viewers are not even required to handle embeddded fonts at all.

Significance for preservation

The use of technical protection technology restricts the (long-term) accessibility of files. Viewing of files may be password-protected, or restricted to certain hardware, software or physical locations. It will also make migration to some alternative format (which may be needed as a preservation action at some point) more difficult, or even impossible.

7.8 Transparency and complexity

Text content in *EPUB* 2 is either in *XHTML1.1* or *DtBook* (which is *XML*) format. In *EPUB* 3 *XHTML5* is used. These are all highly transparent and both human- and machine-readable. These formats are also relatively simple. The same is true for the Package and Navigation documents. This transparency disappears for publications that contain encrypted resources. The list of Core Media Types also includes a number of formats that are less transparent and more complex (e.g. image formats, fonts, audio and video), but these are all common formats that are widely supported by existing software, including (open source) programming libraries. *EPUB*'s mechanism for defining and locating the resources inside each publication is also straightforward. Finally, the use of *ZIP* as the container format does add some complexity (and reduces overall transparency), but the ubiquity of *ZIP* means that the contents of an *EPUB* publication can be inspected by simply opening it with any *ZIP* tool.

Summarising, *EPUB* is a fairly transparent format, provided that it doesn't contain any encrypted resources. The formats used for text content and metadata are relatively simple, although the image, audio and video formats are considerably more complex (and less transparent).

Significance for preservation

Formats that are transparent are generally easier to migrate to some other format, which may be needed as a preservation action at some point. Complex formats can be more difficult (and costly) to manage and preserve.

7.9 External references

EPUB 2

All Publication Resources of an *EPUB* 2 publication must be located within the *EPUB* container. *EPUB* 2 publications may contain references to fonts that are not embedded, and if these fonts are not locally installed on the user's PC this may affect the appearance of these files. On a side note, even though *EPUB* 2 permits font embedding, fonts are not included in the list of Core Media Types. As a result, *EPUB* 2 viewer applications are not required to be capable of handling embedded fonts.

EPUB 3

Section 5.3 of *EPUB Publications 3.0* states that all Publication Resources must be located within the *EPUB* container, with the exception of audio and video resources, which may be located either in the Container or remotely [6]. In the latter case, such resources are known

as *Remote Resources*. Their presence can be identified via the *manifest* in the Package Document [15].

Like *EPUB* 2, *EPUB* 3 publications may contain references to non-embedded fonts, which may affect appearance if these are not locally installed on the user's PC. Unlike *EPUB* 2, Fonts *are* included in the Core Media Types, so embedding should always result in conforming readers to display hem correctly.

Significance for preservation

File objects that contain external references are not self-contained, making their rendering, appearance or functionality dependent on external resources. If the (links to the) external resources are lost, such files may not render as originally intended.

7.10 Authenticity: digital signatures

EPUB (both 2 and 3) supports the use of digital signatures. Like an ordinary signature, the purpose of a digital signature is to demonstrate that a file was created by the person who signed it, and that no changes were made to it afterwards. Digital signatures typically work in the following way. First, a *hash function* is applied to the raw byte data in a file, which produces a so-called *hash value* or *message digest*. This is a fixed-size sequence of bits. The important thing to remember here is that hash functions work in such a way that even the slightest change in their input (here: the data in the file) immediately result in drastic changes in their output. As a next step, the message digest is encrypted using a private key that is known only to the owner or creator of the original document. The encrypted message digest serves as the signature, and is stored. A recipient (or user) of the file can then verify its integrity using a signature verifying algorithm. Given the file's contents, a public key and the signature, the signature verifying algorithm checks whether the message digest as it is computed from the received file matches the signature. If this is the case, this substantiates the file's authenticity.

By itself this does not warrant the identity of the owner of the public key. In other words: the signer of a document may pretend to be someone else. Users of digital signatures may therefore register their public key with a certificate authority, which is a third party that issues certificates that contain a user's public key and information on his or her identity [30].

In *EPUB*, signatures can be created for the publication as a whole, or for individual resources inside it. Signatures are stored in the (optional) *signatures.xml* resource in the *META-INF* directory. A detailed discussion of signatures in *EPUB* can be found in Section 2.5.6 of [8].

Significance for preservation

Digital signatures provide a mechanism for ensuring that a document is authentic and hasn't been tampered with. This is particularly important in the case of e.g. legal documents.

---

[15] Manifest items that are remote resources have a *remote-resources* property.

7.11 Re-usability

Because text content is stored as either *XHTML* or *XML*, the logical structure of an *EPUB* file is explicitly defined. This would make migration to some other XML-based format straightforward, and as a result the contents of an *EPUB* publication are highly re-usable.

Significance for preservation

Re-usable formats are easier to migrate to some other format, which may be needed as a preservation action at some point.

7.12 Summary

Table 8 summarises the main findings of this chapter, itemized by *EPUB* version. It also shows the functionality-related aspects that were discussed in Chapter 5.

	EPUB 2	EPUB 3
Preservation		
Ubiquity, viewer support	Good (but current e-book market dominated by proprietary formats!)	Poor (no viewer support whatsoever as of May 2012)
Disclosure	Open, specifications freely available	Open, specifications freely available
Documentation quality	Good	Good
Stability	Major changes between EPUB 2 and 3	Major changes between EPUB 2 and 3
Support identification tools	Good (*File*, *Tika*); poor (*DROID*, *FIDO*); also no registered identifiers (MIME / PUID)	Good (*File*, *Tika*); poor (*DROID*, *FIDO*); also no registered identifiers (MIME / PUID)
Support validation tools	Two tools exist, but results not in complete agreement. Handling of output problematic in workflows	One tool exists. Handling of output problematic in workflows
Support feature extraction tools	Poor (no detection of DRM, encryption)	Poor (no detection of DRM, encryption)
Intellectual Property Rights	No patents / licensing restrictions	No patents / licensing restrictions
Technical protection	DRM / encryption possible	DRM / encryption possible
Transparency	Good in absence of DRM / encryption	Good in absence of DRM / encryption
Complexity	Low for text and metadata; images and ZIP container increase overall complexity	Low for text and metadata; images, audio, video and ZIP container increase overall complexity
External references	Not allowed, but references to non-embedded fonts are possible	References to external audio / video and non-embedded fonts are possible
Authenticity	Support for digital signatures	Support for digital signatures
Re-usability	Good (XML / HTML content highly re-usable)	Good (XML / HTML content highly re-usable)
Functionality		
Advanced layout / typography	No	Yes
Fixed layout	No	Yes (optional)
Scripts	No	Yes (Javascript)
Multimedia (audio, video)	No	Yes

Table 8 Assessment of *EPUB* 2 and 3: summary of findings related to preservation and functionality.

8 Conclusions and recommendations

This chapter wraps up the main conclusions of this report. It also provides some recommendations.

8.1 Strengths

EPUB has a number of strengths that make it attractive for preservation. It is an open format that is well documented, and there are no known patents or licensing restrictions. It is largely based on well-established and widely-used standards such as *ZIP*, *XML* and *XHTML*. Consequently, the format scores high marks for transparency and re-usability. For situations where authenticity is crucial (e.g. legal documents) all or parts of a document can be digitally signed. Also, *EPUB* 2 is a popular format with excellent viewer support, including several open source implementations.

8.2 Concerns

Despite these strengths, the following observations raise some concerns about *EPUB*'s suitability for preservation.

Dominance of proprietary formats

Even though *EPUB* is the most popular *vendor-independent* e-book format, its role is nevertheless fairly limited because the current e-book market is dominated by proprietary formats.

Stability over time

EPUB 3 shows quite major changes relative to version 2, which raises concerns about the format's stability over time. These concerns are reinforced by the fact that *EPUB* 3 is heavily dependent on *(X)HTML5* and *CSS3*, both of which are unfinished "works in progress", which may undergo various changes before being finalised.

No viewer support for *EPUB* 3

At this moment (May 2012) viewer support for *EPUB* 3 is still virtually non-existent. Because of this, *EPUB* 3 is not recommended as an archival format at this point. However, *EPUB* 3 is still in its early stages, and viewer support is likely to improve soon.

Limited support by characterisation tools

The format is not optimally supported by existing characterisation tools. Recent versions of *Unix File* (5.11) and *Apache Tika* (1.2) correctly identify *EPUB* files, but *DROID* and *FIDO* only identify at the level of the container format (*ZIP*). Two validator tools exist, but currently only one of them supports *EPUB* 3. Both tools have output handlers that are not well suited for use in automated workflows. To my knowledge no feature extraction tool exists that is capable of extracting information on aspects such as digital rights management and encryption, both of which are essential within a preservation context.

Impact of digital rights management and encryption

The possibilities for using digital rights management and encryption are a potential threat to the (long-term) accessibility of *EPUB* files. This is exacerbated by the fact that existing feature extraction tools do not detect the presence of such features.

External references in *EPUB* 3

Starting with version 3, *EPUB* files may contain references to audio and video resources that are stored remotely (i.e. not inside the *EPUB* container). Files that depend on such *Remote Resources* are not self-contained, and their functionality may not persist over time.

Foreign Resources

Both *EPUB* 2 and 3 have a list of *Core Media Types*, which are file formats that *must* be supported by any *EPUB* viewer. The use of other formats is allowed, but viewer applications are not required to render them. For such *Foreign Resources*, a fallback to a file that *is* a Core Media Type must be defined. This means that such resources have multiple representations, and the representation that is shown to the user (or converted in case the publication is migrated to some other format) will be system-dependent. This introduces a degree of unpredictability that is not desired in an archival document.

Scripts in *EPUB* 3

Starting with version 3, *EPUB* files may contain *Javascript* resources. This introduces a number of potential security and privacy risks, and repositories have to decide how to deal with these.

EPUB 2 not suitable for all content

Finally, not all types of content can be adequately represented in *EPUB*. *EPUB* 2 was primarily developed with "general audience books" (e.g. fiction) in mind. It lacks the advanced layout and typographic capabilities that are needed for publications that are heavily dependent on advanced layout and typography. Examples are scientific and technical publications, publications that contain mathematical equations, textbooks and comic books. Also, it does not support *fixed-layout* documents (i.e. documents with a page layout that is unaffected by the viewing device or viewer settings, similar to *PDF*). The *EPUB* 3 specification should overcome these limitations, but viewer support for *EPUB* 3 is nonexistent at this stage. However, as the *EPUB* 3 specification was only released by the end of 2011, this may well improve soon. Also, *EPUB* 2 files may contain embedded fonts, but conforming readers are not required to handle them. This implies that such files may not display as originally intended. Because of all this, *EPUB* is mainly suitable for simple text-centred publications without any sophisticated layout or typography requirements at this stage. This may change as soon as adoption and viewer support for *EPUB* 3 start improving.

8.3 Recommendations: *EPUB* for archival storage

Following the conclusions of this report, a number of recommendations can be made on the acceptance of *EPUB* files for archival storage:

1. For now[16], do not accept *EPUB* 3 publications, until adoption and viewer support have improved.

2. Do not accept *EPUB* publications that contain digital rights management and encryption features. Encrypted (obfusciated) fonts may be an exception (although conforming *EPUB* 2 readers are not even required to handle embedded fonts at all). These features can be detected from the (presence of the) *rights.xml* and *encryption.xml* resources in the *META-INF* directory.

3. Do not accept *EPUB* publications that contain resources that are not on the list of *Core Media Types* (i.e. *Foreign Resources*). These can be detected from the presence of a *fallback* attribute on the manifest item element that represents the resource in the *Package Document*.

4. Do not accept *EPUB* publications that contain *Remote Resources*[17]. These can be detected from the presence of the *remote-resources* property on the manifest item in the *Package Document*.

5. Be alert for *EPUB* publications that contain *Javascript* resources[18]. These can be detected from the value of *media-type* on the the manifest item in the *Package Document*. For *Javascript* resources, *media-type* is set to *text/javascript*.

6. Since *EPUB* 3 no longer uses the *DtBook* format as an alternative to *XHTML*, publications that contain *DTBook* resources should be avoided. This can be detected from the value of *media-type* on the the manifest item in the *Package Document* (which should *not* be *application/x-dtbook+xml*).

7. Use *Epubcheck* or *FlightCrew* to verify if *EPUB* files are in accordance with the format's specifications (but keep in mind that both tools may disagree!).

8. For now[19], do not use *EPUB* for publications that are heavily dependent on advanced layout and typography. Examples are scientific and technical publications, publications that contain mathematical equations, textbooks and comic books. Once the adoption and support of *EPUB* 3 have improved, this recommendation may need to be revised, as *EPUB* 3 should overcome most of *EPUB* 2's current limitations in this regard.

8.4 Community recommendations

The following recommendations are primarily directed at the digital preservation community as a whole:

---

[16]July 2012

[17]Currently only possible for audio and video content in *EPUB* 3

[18]Only possible in *EPUB* 3

[19]July 2012

1. Add the *EPUB* format to existing file format registries (e.g. *PRONOM*) and create unique identifiers for versions 2 and 3 of the format.

2. Consider initiating a registration procedure for *EPUB* at the *Internet Assigned Numbers Authority* [3].

8.5 Tool recommendations

The following recommendations are mainly aimed at developers who are wishing to contribute to improved characterisation of *EPUB*:

1. Create *EPUB* file signatures for *DROID* and *Fido*.

2. Get involved in the development of *Epubcheck*, and/or *FlightCrew*, and write an alternative (*XML*-based) output handler.

3. Develop a feature extraction tool that goes beyond what *epub-tools* and *Tika* currently do (which only extracts information from the *Package Document*). This should include any information on digital rights management, encryption, and digital signatures (all stored in the *META-INF* directory). Alternatively (and perhaps even better), extend *Epubcheck* and/or *FlightCrew* to report this information (most of which they must be analysing already for the validation).

9 References

[1] Open eBook [Internet]. Available from: http://en.wikipedia.org/wiki/Open_eBook

[2] International Digital Publishing Forum [Internet]. Available from: http://idpf.org/

[3] MIME Media Types [Internet]. Available from: http://www.iana.org/assignments/media-types/index.html

[4] EPUB [Internet]. Available from: http://en.wikipedia.org/wiki/EPUB

[5] Zip (file format) [Internet]. Available from: http://en.wikipedia.org/wiki/ZIP_file_format

[6] EPUB Publications 3.0, Recommended Specification [Internet]. 2011. Available from: http://idpf.org/epub/30/spec/epub30-publications-20111011.html

[7] EPUB Content Documents 3.0, Recommended Specification [Internet]. 2011. Available from: http://idpf.org/epub/30/spec/epub30-contentdocs-20111011.html

[8] EPUB Open Container Format (OCF) 3.0, Recommended Specification [Internet]. 2011. Available from: http://idpf.org/epub/30/spec/epub30-ocf-20111011.html

[9] EPUB Media Overlays 3.0, Recommended Specification [Internet]. 2011. Available from: http://idpf.org/epub/30/spec/epub30-mediaoverlays-20111011.html

[10] ANSI/NISO Z39.86 Specifications for the Digital Talking Book [Internet]. 2005. Available from: http://www.niso.org/workrooms/daisy/Z39-86-2005.html

[11] Open Packaging Format (OPF) 2.0.1 v1.0.1, Recommended Specification [Internet]. 2010. Available from: http://idpf.org/epub/20/spec/OPF_2.0.1_draft.htm

[12] Kasdorf B. EPUB 3: Not Your Father's EPUB. Information Standards Quarterly [Internet] 2011. 23:4–11. Available from: http://www.niso.org/publications/isq/2011/v23no2/kasdorf

[13] EPUB 3 Fixed-Layout Documents [Internet]. 2012. Available from: http://idpf.org/epub/fxl/epub-fxl-20120313.html

[14] How Publishers Should Prepare for EPUB 3 [Internet]. Available from: http://www.digitalbookworld.com/2012/how-publishers-should-prepare-for-epub-3/

[15] EPUB 3 Sample Documents [Internet]. Available from: http://code.google.com/p/epub-samples/

[16] DROID, Digital Record Object Identification [Internet]. Available from: http://sourceforge.net/projects/droid/

[17] FIDO - Format Identification for Digital Objects [Internet]. Available from: http://www.openplanetsfoundation.org/software/fido

[18] Fine Free File Command [Internet]. Available from: http://www.darwinsys.com/file/

[19] EpubCheck [Internet]. Available from: http://code.google.com/p/epubcheck/

[20] Morelli D. epub-tools [Internet]. Available from: http://ui3.info/d/proj/epub-tools.html

[21] Brown A. Digital Preservation Guidance Note 1: Selecting file formats for long-term preservation [Internet]. 2008. Available from: http://www.nationalarchives.gov.uk/documents/selecting-file-formats.pdf

[22] Sustainability Factors [Internet]. Available from: http://www.digitalpreservation.gov/formats/sustain/sustain.shtml

[23] EPUB, Electronic Publication, Version 2 [Internet]. Available from: http://www.digitalpreservation.gov/formats/fdd/fdd000278.shtml

[24] EPUB, Electronic Publication, Version 3 [Internet]. Available from: http://www.digitalpreservation.gov/formats/fdd/fdd000308.shtml

[25] Comparison of e-book formats [Internet]. Available from: http://en.wikipedia.org/wiki/Comparison_of_e-book_formats

[26] DeMott A. epub Reader Software [Internet]. Available from: http://www.jedisaber.com/eBooks/Readers.shtml

[27] Readium: Digital Publishing meets Open Web [Internet]. Available from: http://readium.org/

[28] W3C Confirms May 2011 for HTML5 Last Call, Targets 2014 for HTML5 Standard [Internet]. Available from: http://www.w3.org/2011/02/htmlwg-pr.html

[29] It's Simple: ePub is Open, Except When It's Wrapped in DRM, And Then It's Not [Internet]. Available from: http://ereads.com/2009/08/its-simple-epub-is-open-except-when-its.html

[30] Digital Signature [Internet]. Available from: http://en.wikipedia.org/wiki/Digital_signature

[31] Apache Tika [Internet]. Available from: http://tika.apache.org/

[32] FlightCrew [Internet]. Available from: http://code.google.com/p/flightcrew/

[33] Frequently Asked Questions for FlightCrew [Internet]. Available from: http://code.google.com/p/flightcrew/wiki/FAQ