PDF files – documents for eternity or not quite yet?

Pfaffhausen, 20th August 2013: *PDF files long ago became the documents of choice for everyday use. Almost all applications support the PDF format. In Windows, Mac, Linux, on your mobile telephone – PDF documents can be displayed anywhere. Displaying PDF documents is rarely a problem and PDF files are therefore readily used in archiving. A practical example demonstrates where difficulties may lie. First, however, a quick look back to the origins of the PDF format and PDF support in ArchivistaDMS.*

Same origin, different effects

As the first version of our DMS solution Archivista came on the market in 1998, the PDF format was already celebrating its fifth birthday, since the **PDF specification was first released in 1993.** Yet its origins go a lot further back; the **Postscript** language – the predecessor of the PDF format – and the first printers associated with it came onto the market in 1984. In contrast to other print languages, Postscript offered the option of freely positioning elements in the print memory and committing these to paper only at the very end. This made Postscript far superior to other print languages that could only be printed out line by line. At the start, Postscript printers were terribly expensive – the cost of a black/white 300dpi printer ran into five figures.

The high prices were due to printer manufacturers having to pay high licence fees in order to be able to bring 100% compatible Postscript printers onto the market. Postscript printers that did not need a licence, but which were nevertheless able to process Postscript commands, began to come on the market for a lot less money. **With**

the help of software (Ghostscript), it quickly became even more cost-effective to convert Postscript files to image files. These could then be printed out using an ordinary, inexpensive laser printer. This practice had one big disadvantage – creating a page could take an unbearably long time. In 1990, it took **around 2 hours to create (scan) a single page on a 386 computer with 4 MB RAM.** And woe betide the user if a problem occurred (98% of the time!), or if there were typos to be taken out… once scanned, the pages could be displayed on the screen relatively easily and printing using a reasonably-priced machine became the norm.

At this time, the first attempts were being made to display Postscript files directly on screen (in real time). These attempts failed, because no-one was prepared to wait two hours for a single page. So, in **1993, a simplified version of Postscript – the PDF format – was released.** As the name suggests, the Portable Document Format made it possible to display documents independently of any particular software or operating system. The **first versions of Acrobat were released by the manufacturer (Adobe) for displaying files using a viewer (not available free of charge).** The files were extremely large since, in the beginning, PDF only contained ASCII characters.

Displaying a scanned logo, for example, could take up an enormous amount of space, even for a compressed Tiff file; image files had to undergo time-consuming conversion into text files and these then **required around 100 times more storage space.** Furthermore, the reliability of PDF files often left a lot to be desired. The PDF files had a habit of leaving out important source file material. Sometimes the created files contained errors or there was a problem with the PDF display programme or the fonts were not available, so the files could not then be opened or were so corrupted that they could no longer be read.

The first version of Archivista came onto the market at precisely this time. And **because with Archivista the idea was that it should have been possible to preserve the accessibility of documents in long term, the PDF format was not considered for the "original file".** The PDF specification changed four times between 1993 and 1998 and now (after 20 years) we are already into double figures. By comparison, Archivista guarantees 30 years readability of the same file. At the moment, the score is 20:11 for the PDF format and 15:1 for Archivista. In other words, the **PDF format has changed 11 times in 20 years** (not including the PDF-X and PDF-A sub-formats) **whilst Archivista hasn't changed once in 15 years.**

Of course, no-one would deny that the use of PDF files is immense, but long-term storage of the documents is no less problematic. With Archivistabox, this problem has been overcome, since **we can always convert PDF files into image files and continually recreate PDF files from the archived data.** But in doing so, we continually experience problem cases, in which customers assume the fault lies with ArchivistaBox.

Acrobat displays PDF with no problems…

For a number of years, we have been archiving data from a widely-used ERP solution, in which the documents are delivered in PDF format. Following the adoption of a new version of the ERP software, we suddenly began encountering problems with the processed PDF files. Around half of the text rows were not correctly aligned (mostly at the left hand edge). Both the **customer and the supplier confirmed that they had no problems displaying the PDF documents in Acrobat.** This resulted in the following email conversation:

```
> May I ask whether you were able to address the problem
encountered by Mr. Hofer?
> We're under a bit of pressure here regarding time, since we've
not been able to archive the "electronic documents" for well over
a month now.
```

> Many thanks for your feedback.

Our research prompted us to respond as follows:

I tried to get hold of Mr. Hofer last week. I wasn't able to come up with a definite result, only a supposition. The supposition – which I've been able to confirm with checks made this evening – led me to believe that the PDF files had not been created properly. But first things first, in other words what steps were taken to arrive at this conclusion using file GU1000439.pdf.

First assumption: Obsolete libraries in ArchivistaBox

Installation of the latest Debian version and associated tools (time needed: 1.25 hours). Result: An error is still present in the PDF file

Second assumption: There is an error in the PDF file

Checking the file with different Acrobat readers: Acrobat 6.0 and 9.x were tested and installed under Windows and Linux respectively. The file was incorrectly displayed with Acrobat 6.0, but correctly displayed with 9.x. The file was however issued with Version 1.4 (and would therefore have to be correctly displayed with Acrobat 5.x), see:

```
pdfinfo GU1000439.pdf
Title: GU1000439.pdf
Author: Oracle Reports
Creator: Oracle11gR1 AS Reports Services
Producer: Oracl e PDF driver
CreationDate: Wed Jul 17 14:39:57 2013
ModDate: Wed Jul 17 14:39:57 2013
Tagged: no
Pages: 1
Encrypted: no
Page size: 595 x 842 pts (A4)
File size: 342918 bytes
Optimized: no
PDF version: 1.4
```

That means the file should also be correctly displayed with Acrobat 6.0, although that is not the case. In order to verify this, I additionally installed the very reliable Foxit PDF viewer. Due to its size, installing Acrobat is somewhat time-consuming (time taken: 1.5 hours).

Third assumption: Error messages must have appeared at some point during the processing

I was unable to find any error messages in the current ArchivistaBox (out-of-date Ghostscript version). So, I installed Ghostscript 9.x in the test environment. This can also be problematic on account of the different libraries (dependencies) and took up 2 hours of my time. However, I was then able to see the following error messages (extracted from the log file when processing with Ghostscript 9.x):

```
Filename: /home/data/archivista/ftp/pdf/GU1000439.pdf
pdf
**** Unknown operator: '–10.16' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–10.20' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–10.20' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–51.44' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–21.40' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–32.08' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–21.40' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–32.08' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–206.88' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–12.36' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–12.40' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–13.04' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–12.36' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–292.00' looks like a malformed number,
replacing with 0.
**** Unknown operator: '–1.40' looks like a malformed number,
replacing with 0.
```

This brought us closer to the problem. The text elements were not positioned correctly and the delivered file was not correctly displayed. Relative positions are often given in PDF files: a text element may, for example, be positioned 1.4 cm left (-1.40). And here, in fact, was the problem – the error message "–1.40" included two minus signs (malformed number). In this respect, Ghostscript is clearly less error-tolerant than Acrobat 9.x. I think this is correct because -1.40 is not equivalent to –1.40. As is often the case, the data is incorrect; a number may be negative, but may not be written with a double minus sign (–). Arriving at this conclusion took an additional 1.25 hours.

Fourth assumption: It ought to be possible to correct the data oneself.

The PDF file was sent out in a compressed format (to save space). At first, I only trusted pdflib to unpack and re-save the PDF file. But once compiled, I discovered that I would need at least a few hours (if not days) to write a programme. So, I looked around and found pdftk with the following options:

```
pdftk GU1000439.pdf output temp.pdf flatten uncompress
```

This transforms a PDF file into the ANSI format, i.e. the minus values can be corrected in the file temp.pdf (and then saved under fixed.pdf). I did this manually first time around to see if it worked (see insert: fixed.pdf). This took another 1.5 hours.

Fifth assumption: This can be done more quickly with a program

Finally, I wrote the program; this can now be used for volume testing (time taken: 0.5 hours).

```
===========================================================
#!/usr/bin/perl
use strict;
use lib qw(/home/cvs/archivista/jobs);
use AVJobs;
my $file = shift;
my $ftmp = "tmp.pdf";
my $fok = "fixed.pdf";
my $cmd = "pdftk $file output $ftmp flatten uncompress";
my $res = system($cmd);
if ($res==0) {
my $content = "";
readFile2($ftmp,$content);
$content =~ s/(–)([0-9]+)(.)([0-9]{2,2})(s)(TD)/-$2$3$4$5$6/g;
$content =~
s/(–)([0-9]+)(.)([0-9]{2,2})(s)(.*?)(s)(TD)/-$2$3$4$5$6$7$8/g;
writeFile($fok,$content,1);
}
===========================================================
```

Note: The above program only removes superfluous double minus signs. It is not able to correct relative file positions. But this does not appear (at first glance) to have any consequences either in Acrobat, XPDF or Foxit. Final comment: In principle, the problem of the double minus signs ought to be solved by the person causing it. The PDF files are clearly not being sent out correctly. If this cannot be put right, we have to install a volume test and a final fix facility. This could easily take up another 10 – 16 hours.

The total time spent on this job so far (including 45 minutes for the creation of this email) amounts to 8.75 hours. I eagerly await some comment as to how we should proceed.

So this was our response, to which in principle nothing need be added. One point should however be made. **The question of whether a problem is a bug does not go far enough in this instance. Problems are there to be solved.** ArchivistaBox customers continue to receive support even where it is clear that a problem has occurred which is not the fault of ArchivistaBox. And we keep on clearly communicating where problems are likely to occur and how (and with what efforts) such problems can be rectified.

Processing has been speeded up by a factor of 1:200,000

We have to face the fact that – even after 15 years of development – PDF documents are still not suitable for long-term archiving. Scanned image files are much more suited to the purpose. There are those who would claim that the scanning process takes too long, so let it be clearly stated that a **single ArchivistaBox is capable of handling up to 2.4 million pages per day (see comments and tips on PDF tools at pro-linux.de)** or 28 pages per second. This is, by the way, around 200,000 times faster than 20 years ago, when a single page took about 2 hours.

It’s great that today’s technology gives us options that 20 years ago would have been unthinkable, but it is also amazing that we are facing the very same problems (one minus sign too many and the file has had it!). As an ArchivistaBox customer, **you can relax on two counts. Firstly, ArchivistaBox customers receive image files of all filed data “on-the-fly” and secondly, we are always on hand to help solve any “problem cases” that may arise.** And, moreover, all data (including PDF files) is truly captured for eternity.