CIP3 application note

Recommendations for embedding CIP3 PPF data in job files.

Version 1.0

History:

October 18, 1997, Darmstadt. Version 1.0 approved by CIP3 Consortium.

Author

Martin Bailey, Harlequin Group plc

Reviewers

John Felleman, Adobe Systems, Inc.

Karel Di Bie, Agfa-Gevaert N.V.

Peter Kastenholz, Linotype-Hell A.G.

David Hornbacher, Ultimate Technographics Inc.

Introduction

The CIP3 consortium has created a vendor independent file format designed to allow transfer of pre-press, press and post-press related data between stages in a print production environment [CIP3]. The consortium has not defined the ways in which this data should be used, but recognizes that some of the workflows expected to be implemented may involve the embedding of PPF data within a PostScript language file which also contains the graphics description of the pages or imposed flats for the job.

In order to allow such workflows to be implemented with minimal risk of incompatibilities, this advice note outlines the recommended method for such embedding.

It also covers the related topics of references to external PPF data files from within PostScript language files, and the embedding or referencing of PPF data in TIFF/IT and PDF files.

Example files and some detailed comments in this document refer to PPF version 2.1, but the methodology described is appropriate to any version of PPF.

Guiding principles

In generating these recommendations the following explicit requirements have been used:

PostScript parsing

The CIP3 specification for PPF describes a file format designed to be parsable using a PostScript language compatible interpreter. Embedding the file into a larger PostScript language file should not affect that.

A single parsing mechanism

As far as possible the same parsing mechanism should be appropriate for use on a separate PPF file, and on PPF data embedded in a PostScript language file.

Printable files

The PostScript language file containing the embedded PPF data should not cause PostScript interpretation errors if sent to a PostScript language compatible interpreter which does not include extensions required to parse the PPF data itself. This includes sending files to PostScript Level 1 printers.

Useful position in file

The PPF data embedded in the file should be placed at a position in the file which allows that data to be extracted at or before a point in the processing of the file which renders it useful whilst the rest of the file is processed.

Extractable data

It should be relatively easy to construct a parser to extract PPF data from the embedding file, without requiring a PostScript interpreter.

Conformance

Embedding the PPF data in a PostScript language file should not require any aspect of the existing PPF specification to be changed, or any additional PostScript language procedures or operators to be defined over and above those required to interpret a separate PPF file.

Good Practice

The recommendation must comply with good practice in programming in the PostScript language.

Mapping Pages to Sheets

The PPF specification describes imposed press sheets, each of which may contain a front, a back or (more commonly) both.

A PostScript language file contains descriptions of pages.

Resolving a mapping of these two entities to each other is not strictly related to how PPF data should be embedded in a PostScript language file, but that resolution is a pre-requisite for embedded PPF data to be useful.

The graphic representation of the pages in the PostScript file will be in one of two forms:

Imposed PostScript

The pages of the job have already been passed through an imposition package, and each ‘page’ in the PostScript file is either the front or the back of a sheet.

The first page in the PostScript data will be taken as producing the front of the first sheet in the PPF data, the second page will be taken as referring to the back of that sheet etc. If a sheet does not contain a front, or does not contain a back, then the next ‘sheet side’ in the PPF data will be used instead.

Unimposed PostScript

The pages in the PostScript file are single folios which have not yet been imposed, a later process in the workflow will perform that imposition.

In this situation NO assumption will be made about the mapping of PostScript pages to sheet sides, until such time as the imposition process has been performed (which may be only at the time of output to, e.g. a plate-setter). The ambiguity of this situation should be flagged to warn applications using the PPF data.

Recommendations

Complete copy

The PPF data should be copied, complete and unaltered, into the PostScript language file. The leading `%!PS-Adobe-3.0` line in the PPF file should be included in the embedded data to avoid potential problems with offsets in PPF Directory constructs in version 2.1 and later.

Self-contained

The PPF data must be completely self-contained and must not refer to variables, procedures etc. defined in the preceding portions of the surrounding PostScript language file.

Detection of PPF-awareness

Detection of whether a PostScript language compatible interpreter is PPF-aware should be based on the presence of a procedure or operator with the name `/CIP3BeginSheet`. The following example code defines a variable called `/PPF-aware` which might be used in later code in the same way that many PostScript language prologs define a variable called `/Level2?` which can be used later where the language level of the RIP should change the bahaviour of the job.

```
/PPF-aware /CIP3BeginSheet where dup { exch pop } if def
```

Skipping PPF data

If the file is being processed in a PostScript interpreter which is not PPF-aware then the PPF data must be skipped. The skipping should be done by a suitable PostScript language code fragment. It is not intended that this document define the exact details of that code fragment, but an example of such a fragment is included in the example of embedded data included below.

Marking embedded data

The PPF data, including the detection of PPF-awareness, and PostScript language code to skip the PPF data should be prefixed with a

```
%%CIP3BeginData: (CIP3Imposed)
```

comment, forming a complete line of the file on its own, with no spaces before the leading '%' character. The trailing colon (`:`) is required, but the `(CIP3Imposed)` flag may be omitted. If no flag is appended the default, `(CIP3Imposed)`, will be assumed.

If the embedding PostScript language file defines pages which do not relate directly to sheet sides then the flag `(CIP3Unimposed)` should be appended to the end of the comment line:

```
%%CIP3BeginData: (CIP3Unimposed)
```

If the embedded PPF data is version 2.1 or later and includes multiple sheets an additional flag and value may be included:

```
%%CIP3BeginData: (CIP3Imposed) (CIP3Sheet) <name>
```

where `<name>` is the `CIP3AdmSheetName` of the first sheet described in the PPF file which is defined by the embedding PostScript language file (as recorded in the `Sheet` structure). If no `(CIP3Sheet)` flag is present then the PostScript file will be assumed to define sheets starting from the beginning of the PPF data.

If the first sheet side defined by the embedding PostScript file is not the front of a sheet, then the first side defined by the embedding job may be noted with the addition of either

(CIP3Front) or (CIP3Back). These flags may be used in addition to the (CIP3Sheet) flag. If neither (CIP3Front) nor (CIP3Back) is present then (CIP3Front) will be assumed.

e.g.*

```
%%CIP3BeginData: (CIP3Imposed) (CIP3Sheet)
                                (Cover = pages 1,2,43,44) (CIP3Back)
```

No mechanism is provided for an embedding PostScript file to define multiple, non-contiguous sides from a PPF file. In such circumstances it is recommended that the PostScript file be divided into multiple files, each of which refers to a single, contiguous, series of sheet sides.

Vendors wishing to encode additional flag information at this point in the file may do so by adding more PostScript format strings to the end of the comment line. Such strings should not start with the CIP3 prefix, but with a vendor specific prefix.

The data should be followed by a

```
%%CIP3EndData
```

comment. This should form a complete line on its own, with no spaces before the leading '%' and nothing following it on the same line.

Note that the exact case of these comments should be followed.

If the code used to skip the PPF data requires additional text after the embedded PPF data itself then that should be included between the %%CIP3EndOfFile comment and the %%CIP3EndData comment.

CIP3 has been registered with Adobe Systems Inc as a prefix for Open Structuring Comments within the PostScript Document Structuring Conventions.

Position in file

The PPF data with its surrounding comments should be included between the %%EndComments and %%EndProlog comments near the beginning of the file (see appendix G of [PostScript] for details of ADSC comments).

If the embedding PostScript language file is not ADSC compliant then the embedded PPF data should be included near the beginning of the file, and outside any save/restore contexts used for individual pages or groups of objects within pages.

Number of sheets

Until version 2.1 of the PPF specification a PPF file could only contain a single sheet, i.e. a maximum of two sheet sides.

A PostScript language file may include an arbitrary number of pages.

There may therefore be a desire to include multiple PPF data sets of version 2.0 or earlier within a single PostScript language file. In such circumstances it is recommended that the PostScript language file be split into multiple files, each of which includes the graphic description of a single sheet in order to relate it to a single PPF data set.

If version 2.1 or later of the PPF specification is being used, then a single, multiple-sheet, data set may be embedded within the PostScript language file.

* This example code has been broken into two lines in order to fit this document, it must be included as a single line in the embedding PostScript file.

PPF Directories

Byte offsets in the PPF directory listing introduced in PPF version 2.1 should refer to the offset from the beginning of the `%!PS-Adobe` line of the embedded PPF data, not from the beginning of the embedding PostScript language file. i.e. if the PPF data from the beginning of the `%!PS-Adobe` line, to the end of the `%%CIP3EndOfFile` comment line is extracted to a separate PPF file the PPF Directory should be correct.

Example of embedded PPF data:

The example file shown in the specification for PPF version 2.1 is included in this example.

```
%!PS-Adobe-3.0
%%Title: (Sample PostScript)
%%Creator: (Application X: LaserWriter 8 8.3.4)
%%CreationDate: (14:25 Friday, September 13, 1996)
%%For: (MartinB)
%%Pages: 1
%%DocumentFonts: Helvetica Courier
%%DocumentNeededFonts: Helvetica Courier
%%EndComments
%%BeginProlog
%%CIP3BeginData:
/CIP3BeginSheet where { pop }{
   save userdict /str 1024 string put
      {
      % loop until end line marker
         {
         % loop to read a single line if
         % too long for the buffer
         mark //currentfile str { readline } stopped
         not {
            exit
         } if
         cleartomark
      } loop
      not {
         % end of file - shouldn't happen
         poppop% the string and mark
         exit
      } if
      (%%CIP3EndData) anchorsearch {
         % found end marker
         pop pop pop % string bits and mark
         exit
      } if
      poppop   % the string and mark
   } loop
   restore
} bind ifelse
%!PS-Adobe-3.0
%%CIP3-File Version 2.1

CIP3BeginSheet
(Sheet structure of CIP3 example) CIP3Comment
/CIP3AdmJobName (TestJob) def
/CIP3AdmMake (Prepress Company) def
/CIP3AdmModel (PC210) def
/CIP3AdmSoftware (The Imposition Program) def
/CIP3AdmCreationTime (1994:08:30 12:17:03) def
/CIP3AdmArtist (Stefan Daun) def
/CIP3AdmCopyright (Copyright by Fraunhofer-IGD, 1995) def
```

```
... Many lines of PPF data omitted

625.0 872.0 0 /regm1 CIP3PlaceRegisterMark
30.0 872.0 0 /regm1 CIP3PlaceRegisterMark
CIP3EndRegisterMarks

CIP3EndBack

CIP3EndSheet

%%CIP3EndOfFile
%%CIP3EndData
userdict begin/dscInfo 5 dict dup begin
/Title(Sample PostScript)def
/Creator(Application X: LaserWriter 8 8.3.4)def
/CreationDate(14:25 Friday, September 13, 1996)def
/For(MartinB)def
/Pages 1 def
end def end
/md 149 dict def md begin/currentpacking where {pop /sc_oldpacking
currentpacking def true setpacking}if

... Many lines of prolog omitted

/currentpacking where {pop sc_oldpacking setpacking}if end
%%EndProlog
%%BeginSetup
md begin

... rest of embedding PostScript file omitted.
```


Referencing external PPF files.

Rather than embedding PPF data directly in a PostScript language file there may be some occasions when including a reference to an external PPF file is more appropriate.

This reference may be done by inclusion of a

```
%%CIP3IncludeData: (filename)
```

comment at the same position in the PostScript file as described for embedded data above.

The filename should be a valid PostScript string which defines a valid PostScript file name. It may or may not explicitly include a PostScript device name. How such a reference is resolved is not defined here; it is expected that in some circumstances normal PostScript file search rules would be applied, but in others more sophisticated mechanisms including pre-specified search directories and fuzzy name-matching may be used.

The `(CIP3Sheet)`, `(CIP3Front)` and `(CIP3Back)` flags may be used with an `%%CIP3IncludeData` comment in exactly the same way as they are described above for `%%CIP3BeginData`. They must follow the filename argument.

Example of external reference

```
%!PS-Adobe-3.0
%%Title: (Sample PostScript)
%%Creator: (Application X: LaserWriter 8 8.3.4)
%%CreationDate: (14:25 Friday, September 13, 1996)
%%For: (MartinB)
%%Pages: 1
%%DocumentFonts: Helvetica Courier
%%DocumentNeededFonts: Helvetica Courier
%%EndComments
%%BeginProlog
%%CIP3IncludeData: (%E%cip3/example.ppf) (CIP3Front)
userdict begin/dscInfo 5 dict dup begin
/Title(Sample PostScript)def
/Creator(Application X: LaserWriter 8 8.3.4)def
/CreationDate(14:25 Friday, September 13, 1996)def
/For(MartinB)def
/Pages 1 def
end def end
/md 149 dict def md begin/currentpacking where {pop /sc_oldpacking
currentpacking def true setpacking}if
... Many lines of prolog omitted
/currentpacking where {pop sc_oldpacking setpacking}if end
%%EndProlog
%%BeginSetup
md begin
... rest of embedding PostScript file omitted.
```

Accessing externally referenced PPF data

It is obviously possible for such an external reference to be extended to include the PPF file directly in the case of a PPF-aware RIP. This can be done with the standard PostScript `run` operator, switched on the same test of PPF-awareness as described above. In this case normal PostScript file search rules would be applied.

```
%%CIP3IncludeData: (%E%cip3/example.ppf)
/CIP3BeginSheet where {
   pop
   (%E%cip3/example.ppf) run
} if
```

Association of PPF data with TIFF/IT files.

The TIFF/IT specification describes a multiple-file format which can describe a single page per file set. Given the pre-existing multiple file structure it is inappropriate to embed PPF data within any one of those files, but it is very appropriate to associate a PPF file with such a file set.

This should be done by adding a `CIP3DataFile` tag (37434) to the Final page IFD of the FP file of the set. The tag data is a string which should be interpreted as a file name. This document does not define the rules by which a file should be found from that name – in most cases the PPF file should be saved in the same directory as component CT, LW etc. files.

The TIFF/IT file set can only describe a single side of a single sheet.

If the referenced PPF file is of version 2.1 or later and includes multiple sheets then a `CIP3Sheet` tag (37435) may be added to the final page IFD. Its value should be a string, defining the `CIP3AdmSheetName` of the required sheet, as listed in the `Sheet` structure of the PPF file. This tag is optional; if it is omitted then the first sheet described in the PPF file should be assumed.

If the referenced PPF file includes both front and back of a sheet then the `CIP3Side` tag (37436) may be added to the final page IFD. Its value should be a byte – 0 for the front of the sheet or 1 for the back. This tag is optional; if it is omitted then the first side described in the PPF file should be assumed (note that this could be the back of the sheet if no front is described).

The TIFF/IT specifications ([TIFF/IT ANSI], [TIFF/IT ISO]) follow the lead set by the original TIFF specification, that tags which a reader does not recognize should be silently ignored. This is the case even for TIFF/IT-P1 files. Inclusion of such a new tag will not, therefore, prevent a TIFF/IT file from conforming to the specification.

Implementors should, however, be aware that some TIFF/IT editing software *may* silently remove these tags from TIFF/IT files because they are not covered by the TIFF/IT specification.

The tags described here have been reserved with Adobe Systems Inc., who maintain the TIFF format.

Embedding and referencing PPF data in PDF files.

Adobe is working on a methodology for inclusion of the data which would be included in a PPF file within a Portable Job Ticket Format file (a part of PostScript 3), but it is also appropriate to consider how current PPF files may be associated with current versions of PDF files.

To avoid confusion it is recommended that the mechanism outlined here be described as embedded CIP3 data in PDF files, and the Adobe methodology be described as PDF formatted CIP3 data.

To embed PPF data in a PDF file a new key should be added to the `Catalog` object of the file. The key should be named `/cip3Data` and its value should be an indirect reference to a PS XObject. All the rules for construction of such XObjects as set out in the Adobe PDF specification [PDF] should be followed.

To refer to an external PPF file in a PDF file a key named `/cip3DataFile` should be added to the `Catalog` object. Its value should be a string which describes the location and name of the external PPF file. It may be a direct or indirect reference. The string should be formatted as described in section 7.2 (File Specification) of [PDF]. As in the case of references from PostScript files, how such a reference is resolved is not defined here, in particular, PDF generating applications should not make assumptions about how relative file names will be interpreted by a process later in the work flow.

No PDF file should include both a `/cip3Data` key and a `/cip3DataFile` key - only one should be used in any file.

If the referenced PPF file is of version 2.1 or later and includes multiple sheets and the PDF file defines only a subset of those sheets then a /cip3`Sheet` key may be added to the `Catalog`. Its value should be a string, defining the `CIP3AdmSheetName` of the first sheet defined by the PDF file, as listed in the `Sheet` structure of the PPF file. This key is optional; if it is omitted then it should be assumed that the PDF file defines one or more sheets starting at the beginning of the PPF file.

If the referenced PPF file includes both front and back of a sheet and the PDF file defines only one side then the /cip3`Side` key may be added to the `Catalog`. Its value should be `/Front` or `/Back`. This key is optional; if it is omitted then the PDF file should be assumed to describe one or both sides of the sheet, starting with the front.

Both `/cip3Sheet` and `/cip3Side` keys may be present together, and both may be used in conjunction with either a `/cip3Data` or a `/cip3DataFile` key.

No mechanism is provided for a PDF file to define multiple, non-contiguous sides from a PPF file. There may also be a desire to include multiple PPF data sets of version 2.0 or earlier within a single PDF file. In both such circumstances it is recommended that the PDF file be split into multiple files, each of which includes the graphic description of a single sheet (or contiguous series of sheets described in a v2.1 PPF file) in order to relate it to a single PPF data set.

`cip3` has been registered with Adobe Systems Inc. as a second class name prefix for use in PDF files.

References:

[CIP3] Specification of the CIP3 Print Production Format (Fraunhofer Institute For Computer Graphics (Fraunhofer-IGD), 1997, Version 2.1)

[PostScript] PostScript Language Reference Manual. (Adobe Systems Inc., 1990, 2nd Edition, ISBN: 0-201-18127-4).

[PDF] Portable Document Format Reference Manual. (Adobe Systems Inc., 1996, Version 1.2, ISBN: 0-201-62628-4).

[TIFF/IT ANSI] Graphic Technology – Prepress digital data exchange – tag image file format for image technology (TIFF/IT). ANSI IT8.8-1993.

[TIFF/IT ISO] Graphic Technology – Prepress digital data exchange – tag image file format for image technology (TIFF/IT). ISO/DIS 12639, 1996.

