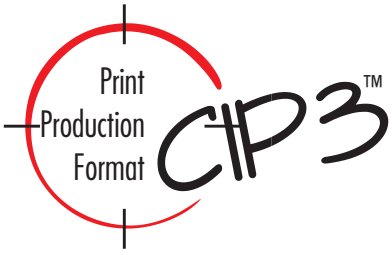
International Cooperation
for Integration of
Prepress, Press, and Postpress

# Specification of the

# CIP3™

# Print Production Format

Version 3.0
June 2, 1998




Stefan Daun
Georg Lucas
Jürgen Schönhut

FRAUNHOFER INSTITUTE FOR COMPUTER GRAPHICS (FRAUNHOFER-IGD)
Document Imaging Department
Rundeturmstraße 6, D - 64283 Darmstadt
Phone: +49 - 6151 -155 220
Fax: +49 - 6151 - 155 299

*Please send any comment on this document to Stefan Daun*
*Email: daun@igd.fhg.de — Fax: +49 - 6151 - 155 299*

**98i008-FIGD**

The CIP3 Print Production Format was specified by the Fraunhofer Institute for Computer Graphics in close collaboration with a group of currently 34 well-known manufacturers. This group is called "International Cooperation for Integration of Prepress, Press, and Postpress" (abbreviated as CIP3). The following companies are members of this group:

- Adobe
- Agfa
- Baldwin Technology Company
- Barco Graphics
- Creo
- Ekotrading-Inkflow
- Eltromat Polygraph
- Ewert Ahrensburg Electronic
- Fujifilm Electronic Imaging
- Gerber
- Goebel
- Graphics Microsystems
- Hagen Systems
- Harlequin
- Heidelberg
- Koenig & Bauer - Albert AG
- Kolbus
- Komori
- MAN Roland
- Mitsubishi Heavy Industries
- Müller Martini
- Nth Degree Software
- Polar-Mohr
- RR Donnelley & Sons Company
- Ryobi
- ScenicSoft
- Scitex
- Screen
- Shinohara Machinery Company
- Toshiba Machine Co., Ltd.
- Ultimate Technographics
- Wohlenberg
- Xerox
- Yamatoya

This document can be obtained from the Fraunhofer Institute for Computer Graphics in Darmstadt in several ways:

1. As Paper Copy
   Send a fax order to Stefan Daun (fax: +49 / 6151 / 155 299) to get a paper copy of the document. You will then be charged a token fee of 100,- DM.

2. Via FTP
   Connect to FTP server *ftp.igd.fhg.de*, log in as user *anonymous* (using your mail address as your password). Then change to the *outgoing/igd_a1/cip3* directory (please note the underline char and the digit one in “igd_a1“), where you can fetch (in binary mode) the document either as a PostScript (*cip3.ps*) or as a PDF file (*cip3.pdf* ). Fetching the electronic document is free.

3. Via WWW
   In the World Wide Web you can find more information about the CIP3 Print Production Format (including the specification document) at the following address:
   `http://www.cip3.org`

# Content

# 1. Introduction

The starting point for the development of the CIP3 Print Production Format (CIP3) was the desire to link printing and the post-printing processes closer to the prepress phase. The various data which is generated in prepress should be compiled to enable a uniform administration and further processing. In this way a repeated, time-consuming data acquisition in the subsequent processing steps should be avoided, because this data had already been available in the prepress phase.

In the first two parts of the study various alternatives for the Print Production Format were investigated. Above all existing standards or rather the de-facto standards were taken into account. It turned out that the formats PostScript and TIFF, which play an important part in the connected sectors, are best suited for a realization of the CIP3. Finally PostScript was chosen as the basic format for the CIP3 format, due to the fact that it is more flexible and can be extended more easily.

In the present report this favorable solution on the basis of PostScript will be presented and the Print Production Format will be specified. With this the format is disclosed and can build the interface to other systems.

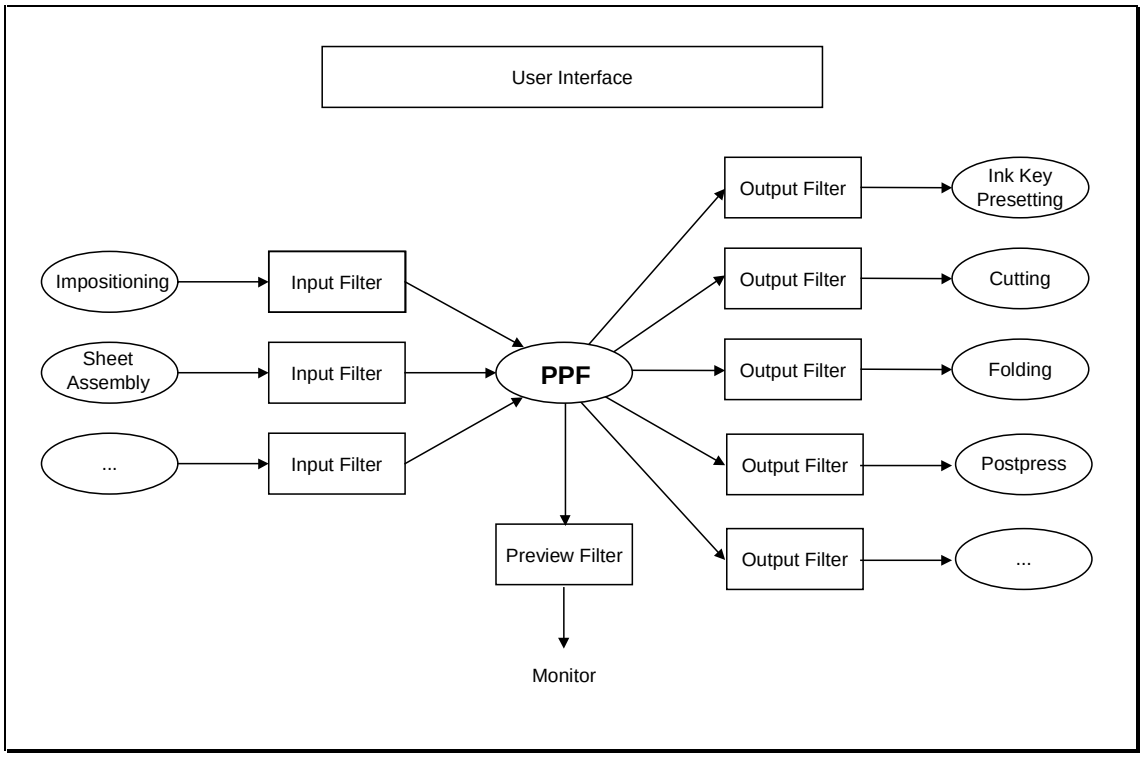



**Illustration 1-1: CIP3-Scenario**

*This illustration shows the conceptual relationship between the components and should not be seen as a description of an implementation.*

# 2. An Overview of the CIP3 Print Production Format

In the ideal case the CIP3 print production format contains all necessary data for print and for the further processing of a job, except for the data screened in high resolution for the printing.

To allow for easy and flexible access the information of the CIP3 format is stored in a structured way. A PPF Directory provides links to the sheets defined in the file. It can be used to directly access a certain sheet. In addition the PFP Directory is used by the product definition section to specify the components that are required to build a certain product.

The information of one single sheet is stored by using Structures, Attributes, and Content. These elements are described in more detail in the following three chapters.

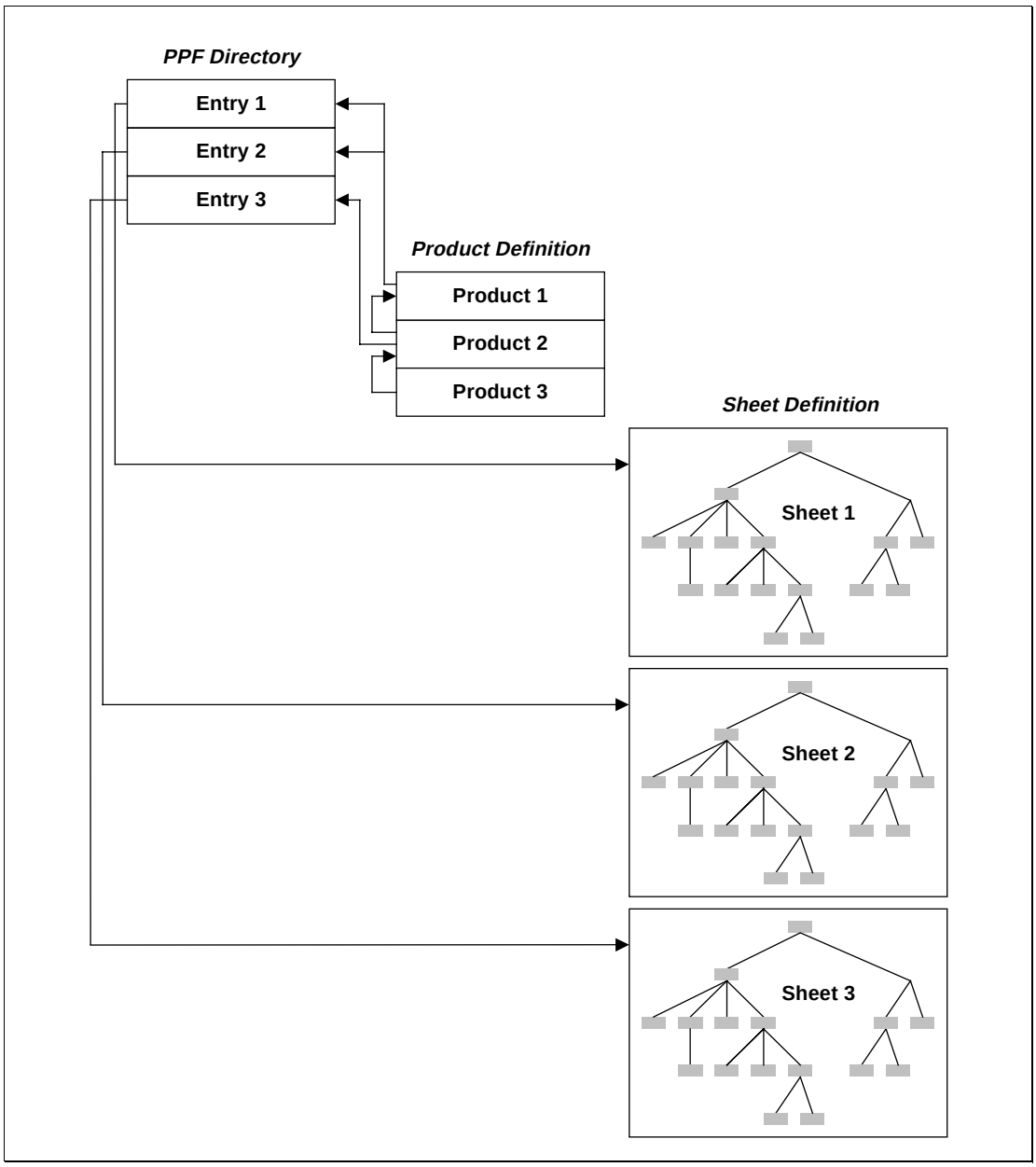



**Illustration 2-1: Building blocks of a CIP3 file**

## 2.1. Logical Structure

Instead of storing the description of a sheet in a flat format, it was decided to provide a structured way of storing the information. Although this prevents the format from being read by a simple “grep“ program, this method allows for much more flexibility.

The structure of a CIP3 sheet builds a tree-like **"part-of-hierarchy"**, in which each subelement is a part of its parent element. If, for example, in the real world a sheet consists of a front and a back, then the corresponding CIP3 sheet description contains a *Sheet* structure, which in turn contains a *Front* and a *Back* structure.

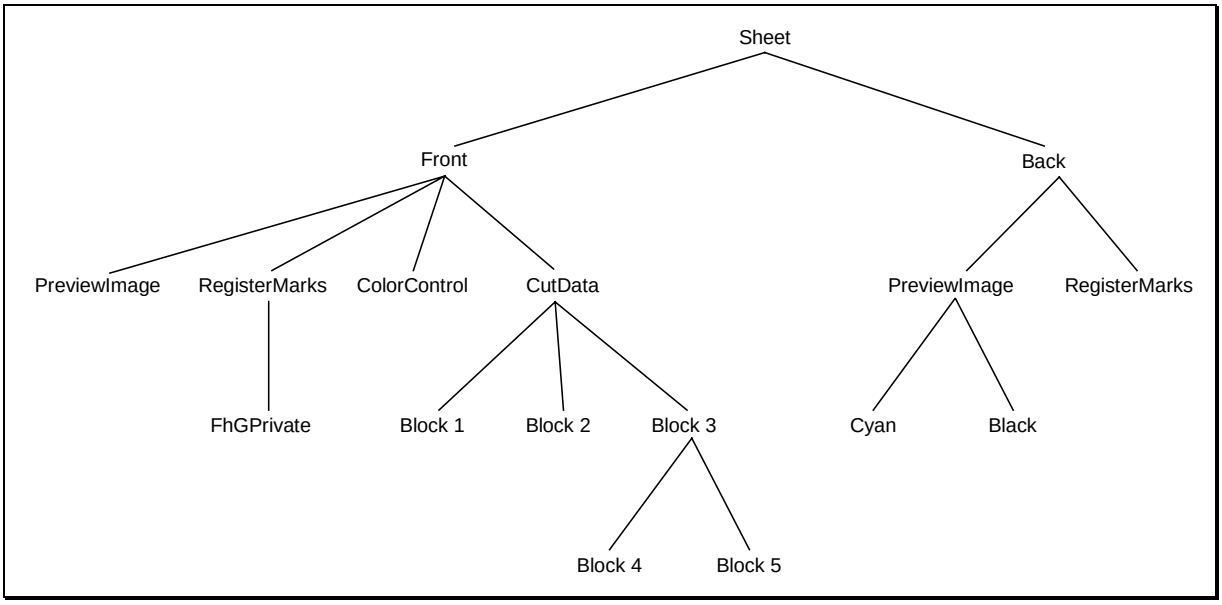



**Illustration 2-1: Example of the hierarchical structure of a sheet within a CIP3 PPF file**

This logical structure is used to establish an **inheritance mechanism**. By this mechanism an element can inherit attributes from its (direct or indirect) parent elements. In every structure element new attributes can be defined or existing attributes can be overwritten.

It is important that only attributes can be inherited. Content portions (like a register mark) cannot be inherited.

Coordinates specified in a structure element are always relative to the origin of this element.

## 2.2. Attributes

Within the CIP3 format attributes are used to store information that characterizes the job. To define an attribute it is necessary to specify the attribute name and its value. Through the definition the attribute is bound to the structure element in which the definition took place. The attribute value is then valid in the structure element, in which the attribute is defined, and in all (direct or indirect) subelements, unless it is overwritten. Attributes can be optional or necessary.

The following data is stored in attributes:

- **Information for administration**
  The name of the sheet creator, date and time of the creation, software with which the sheet has been created etc., are included here.
- **Data for calculating the ink consumption**
  Here it is possible to store two characteristic curves of the transfer, one for the process of copying the data onto film, and one for copying data onto the printing plate.
- **Folding data**
  Data which describe the position of the sheet and the sequence of the folding procedure. This data can be bound to a cut block.

## 2.3. Content

For the first realization the following content types are planned:

- **Continuous tone image with reduced resolution**
  Display of the sheet to be printed with e.g. 1280 * 1024 pixels and 8 Bit depth per color separation. Above all this picture is used for display. By this means it should be possible to check the completeness of the print job and (at least roughly) the position of the cutting and folding edges etc. Moreover, it is used for calculating the ink consumption. For the first approach we have in mind the storage of the separation in CMYK. For the future the CIP3 format will also support composite color. However, at a later stage the use of CIE-L*a*b* might be possible. Then the continuous tone image could be available in either only one of the two types or even in both types simultaneously.
- **Register marks**
  Data which describes the position and type of register marks.
- **Color and ink control fields**
  Data for the description of measuring strips with a number of test fields for the quality measurement. Two basic types of measuring fields are supported: color measuring fields and density measuring fields. Control strips can be build up from these two basic fields.
- **Cutting data**
  Data which describe the cutting blocks.
- **Comments**
  Any comments.

## 2.4. Encoding in PostScript

PostScript is used as the basic format for the CIP3 print production format.

As far as sensible all data which should appear on the paper during a test print is directly coded in PostScript with CIP3.

All **structure elements** of the CIP3 are enclosed by a pair of "**`CIP3Begin...`**" and "**`CIP3End...`**". In order to build the "part-of-hierarchy" it is necessary to nest the structure elements. Each structure element is represented by a PostScript dictionary , which stores all attributes defined in the element.

```
CIP3Begin<Structure1>
<attribute definitions and content of Structure1>
CIP3Begin<Structure2>
<attribute definitions and content of Structure2>
CIP3End<Structure2>
<more attribute definitions and content of Structure1>
CIP3End<Structure1>
```

An **attribute** is defined by use of the PostScript "**def**" operator. Therefore the specification of the name and the value of the attribute is required:

```
/attribute-name value def
```

It is important that attributes must occur in a structure before the first content element and before the next “**CIP3Begin...**”.

For more information about the encoding in PostScript see chapter 3.1.

## 2.5. Usage of the CIP3 Print Production Format

The information stored in a CIP3 file can be used in different ways:

1. It can be parsed by a PostScript interpreter to extract specific information. This is the main purpose of the CIP3 print production format. For example, the following information can be extracted from a CIP3 PPF file:
    - The image data and the transfer curves can be used to calculate the ink consumption and profile and to preset the printing press.
    - The cutting data can be used to automatically generate programs for a cutting device.
    - The folding data can be used to automatically generate programs to control or preset a folding device.
    - The product definition data can be used to automatically set up postprocessing equipment, like gathering or binding machines.
2. It can be printed on a normal PostScript printer. This requires an appropriate PostScript prolog. Depending on the prolog one or more pages per sheet will be printed, which can be used for documentation purposes or to roughly check the completeness of the print job.
3. It can be parsed by a PostScript interpreter to produce a description of the job. This can be a page containing administration data etc.
4. It can be edited to modify or store information in the CIP3 PPF file. By using Private Data (see chapter 3.12) and Private Content (see chapter 3.13) it is even possible to store vendor- or customer-specific information inside the CIP3 PPF file.

It is important that a PostScript interpreter is necessary to parse the CIP3 file. This is mainly due to the fact that the information is stored in the CIP3 file in a hierarchically structured way. But, in order to reduce the complexity of the parsing process, some restrictions in using the PostScript language have been included in the CIP3 PPF specification (see chapter 3.1.2).

# 3. Specification

In the following the Print Production Format has been specified. The following notation is valid:

`Courier normal`: Content of a CIP3 file. The text given in this way must appear in the PPF file in the same way.

*`Courier italic`*: Text given in this way must be replaced by a corresponding figure, name or similar.

*`<Courier>`*: At this place the information described in angle brackets (*< >*) must be supplied.

`...`: Omission sign.

## 3.1. Specification Concepts

### 3.1.1. The CIP3 File Structure

A CIP3 file is structured in a two line heading, the body (containing structures, attributes and content), and an End-Of-File line. The first two lines serve for a definite detection of a CIP3 file. A third, optional, header line may be included in the beginning of a CIP3 PPF file. It contains four characters with codes greater than 128. By this system utilities and other programs will more likely identify the CIP3 PPF file as binary file.

The line should consist of the following characters:

| Char: | % | Ô | Ò | ¤ | Ë |
|---|---|---|---|---|---|
| decimal: | 37 | 226 | 227 | 207 | 211 |
| hex: | 25 | E2 | E3 | CF | D3 |

The three header lines must look like this:

```
%!PS-Adobe-3.0
%%CIP3-File Version 3.0
%ÔÒ¤Ë
```

Please note, that the text in the header lines is case sensitive (i.e. upper/lower case is important). At the beginning of the lines in front of the percent characters no other characters (such as blanks or tabulators) may appear.

The last line serves for the control of the complete transmission of a CIP3 file and must always look like this:

```
%%CIP3EndOfFile
```

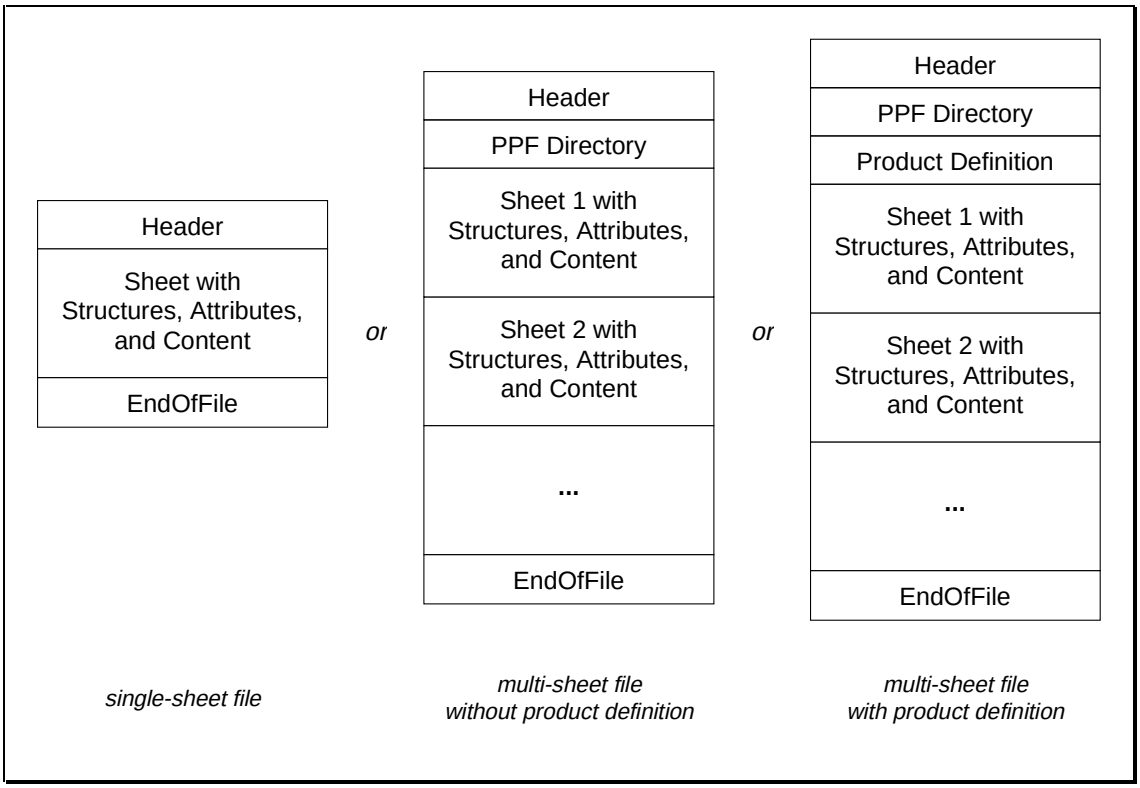



**Illustration 3-1: Syntactical structure of the CIP3 file**

### 3.1.2. Syntax and Data Types

Since the CIP3 Print Production Format is currently encoded in PostScript, the syntax of the CIP3 format follows the syntax rules of the PostScript language. This is especially true for the coding of numbers, names, character strings etc. . Upper/lower case of names and keywords must be observed.

Any occurrence of the "%" character outside a string introduces a comment. The comment consists of all characters between the "%" and the next newline or form-feed character.

In contrast to "real" PostScript, the following rule is established in order to reduce the complexity of the CIP3 format:

> ***A CIP3 file should contain nothing else executable in the CIP3 file body except what is defined in the specification (comments are allowed).***

In particular this means, that it is not allowed to calculate the values of an attribute by means of the PostScript language (e.g. by using operators like mul or add). In addition the implicit version of specifying arrays (using "[" and "]") and dictionaries (using "<<" and ">>") must be used.

The following table lists all PostScript operators that may be used within a CIP3 PPF file. Please note that since version 2.1 of the CIP3 PPF the PostScript operators "begin", "end", and "dict" are no longer supported.

| PS Operator | Meaning |
|---|---|
| [ | begin array |
| ] | end array |
| << | begin dictionary |
| >> | end dictionary |
| Def | associate name with value |

**Table 3-1: List of PostScript operators allowed in CIP3 PPF**

The following chapters show the PostScript data types (see chapters 3.2 and 3.3 of the PostScript Language Reference Manual [PostScript]) which may be used within CIP3.

#### 3.1.2.1. Boolean

A boolean value can be "true" or "false".

**Example 3-1: Boolean Values**

```
true
false
```

#### 3.1.2.2. Integer

An integer object represents mathematical integers within an implementation dependent interval centered at zero. A typical range is $-2^{31}$ to $2^{31}-1$. An integer consists of an optional sign followed by one or more decimal digits.

**Example 3-2: Integer Values**

```
-12
0
+345
1344
```

#### 3.1.2.3. Real

A real object represents a mathematical real number within a much larger interval but with limited precision. A real consists of an optional sign followed by one or more decimal digits with an embedded period (decimal point), a trailing exponent, or both. The exponent, if present, consist of "E" or "e" followed by an optional sign and one or more decimal digits.

**Example 3-3: Real Values**

```
-345.
-3.62
-.01
0.0
4E-2
34.5
155.6e17
```

#### 3.1.2.4. Number

There are two types of numbers provided: integer and real. Throughout this document, number means an object whose type is either integer or real.

**Example 3-4: Number Values**

```
-345.
-3
3.0
4E-2
```

#### 3.1.2.5. Name

A name object can consist of any character except delimiters ("(", ")", "<", ">", "[", "]", "{", "}", "/", "%") and white space (null, tab, line-feed, form-feed, carriage return, space) that cannot be interpreted as a number.

A "/" introduces a literal name. The slash is not part of the name itself, but is a prefix indicating that the following name is literal.

The maximum length of a name is 127 characters.

Within the CIP3 Print Production Format literal names are often used to specify a value from a set of predefined values (like an enumeration type). These predefined values are subject to registration.

**Example 3-5: Name Objects**

```
ANTON
printing
5b
a-z
www.cip3.org
/Left
/Binary
/CutBlock
```

#### 3.1.2.6. String

A string consist of up to 65535 characters enclosed in "(" and ")". Any characters may appear in the string other than "(", ")", and "\", which must be treated specially. Balanced pairs of parentheses in the string require no special treatment. Within a string, the backslash character "\" is treated as an escape character:

| | |
|---|---|
| \n | line-feed or (newline) |
| \r | carriage return (CR) |
| \t | horizontal tab |
| \b | backspace |
| \f | form-feed |
| \\ | backslash |
| \( | left parenthesis |
| \) | right parenthesis |
| *ddd* | character code *ddd* (octal) |

Any string within the CIP3 PPF may be encoded in Unicode. If it is encoded in Unicode, the first two bytes of the string must be the Unicode Byte Order marker, <FE FF>. The high-order byte of a Unicode character appears first in a string.

A non-Unicode String can be easily converted into a Unicode encoded string. The resulting string must start with the Unicode Byte Order marker. Then each character of the non-Unicode string is coded by two bytes: the first bytes is a null byte, while the second byte contains the original non-Unicode character code.

**Example 3-6: String Objects**

```
(This is a string)
()
(This is (OK))
(This \(too\))
(a line\n)
(\376\377\000U\000n\000i\000c\000o\000d\000e)
```

#### 3.1.2.7. Array

An array is enclosed in "[" and "]". Each element of the array may be of a different data type (i.e. one of Boolean, Integer, Real, Name, String, Array, or Dictionary). Arrays may be nested. The maximum number of elements in an array is 65535.

**Example 3-7: Array Objects**

```
[ 1 2 3 ]
[/Left 23.0 true]
[[1] [] ]
[(str) <</a 5>>]
```

#### 3.1.2.8. Dictionary

A dictionary is enclosed in "<<" and ">>" and consists of key-value pairs. Within CIP3 PPF the key must always be a literal name, while the value can be of one of the data types supported by CIP3 PPF (i.e. one of Boolean, Integer, Real, Name, String, Array, or Dictionary). Dictionaries may be nested. The maximum capacity of a dictionary is 65535 key-value pairs.

**Example 3-8: Dictionary Objects**

```
<<
/5b 5.6
/Date (16.01.97)
/arr [1 (str)]
<< /a 0.2 >> >>
```

### 3.1.3. Coordinate System

The default coordinate system of PostScript is used as the coordinate system for the CIP3 format. The origin of this system is in the bottom left corner (of the PostScript coordinate system, see below), the unit is 1/72 inches in both directions of axis.

For the different prepress, press, and postpress processes that are encountered during the life cycle of a CIP3 file it is required to map the coordinates of the PostScript coordinate system to other coordinate systems. Therefore transformation matrices can be defined within CIP3, that specify the mapping between two coordinate systems. Furthermore the extent of each coordinate system can be defined.

The default value for an unspecified transformation matrix is the identity transformation. The default value for an unspecified extent is the value inherited from the source coordinate system. Only the specification of the CIP3AdmPSExtent is required.

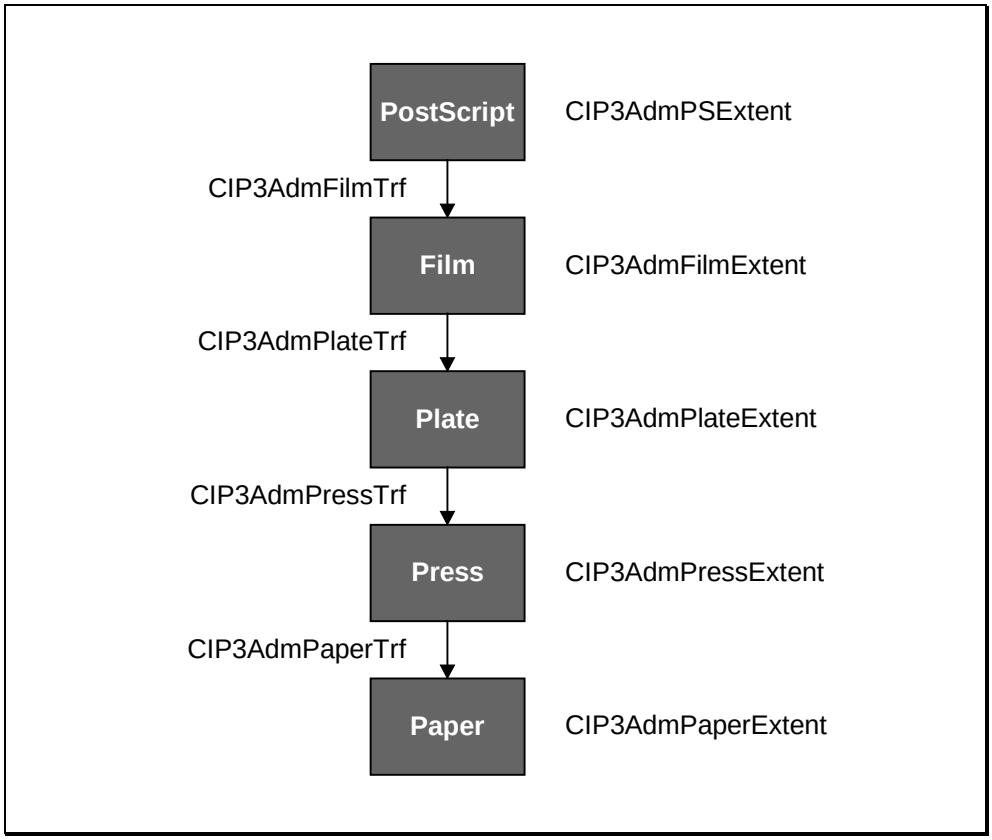



**Illustration 3-2: Coordinate Systems and Coordinate Mapping**

In addition to the native PostScript units it is also possible to use predefined measuring units. If a number is stated without unit, then 1/72 inch is the accepted unit. The measurements listed in the following can be used, by stating the number and - separated by a blank - the abbreviation of the measuring unit.

| Measuring unit | Abbreviation | Conversion factor |
|---|---|---|
| Millimeter | mm | 72/25.4 |
| Centimeter | cm | 72/2.54 |
| Inch | inch | 72 |
| Point | point | 1 |

**Table 3-2: Units in the CIP3 format**

The *point* unit mentioned above is the PostScript point, which is defined as the 72nd part of an inch.

### 3.1.4. Defining the Logical Structure

The logical structure of the CIP3 file is created by PostScript commands of the form "**CIP3Begin*Unit***" and "**CIP3End*Unit***". In general *unit* specifies both type and name of the structure. But since more than one Separation, CutBlock, or PrivateData structure may exist at the same hierarchy level, it is necessary to allow for user defined names for these structures. All substructures of a given structures must have distinct names.

The "**CIP3Begin*Unit***" and "**CIP3End*Unit***" commands need not be on a separate line or at the beginning of a new line.

| Definition | Structure Type | Structure Name |
|---|---|---|
| CIP3BeginSheet<br>CIP3EndSheet | Sheet | Sheet |
| CIP3BeginFront<br>CIP3EndFront | Surface | Front |
| CIP3BeginBack<br>CIP3EndBack | Surface | Back |
| CIP3BeginPreviewImage<br>CIP3EndPreviewImage | PreviewImage | PreviewImage |
| CIP3BeginSeparation<br>CIP3EndSeparation | Separation | *implicitly inherited from the CIP3AdmSeparationNames attribute defined in the Sheet, Front, or Back structure* |
| CIP3BeginRegisterMarks<br>CIP3EndRegisterMarks | RegisterMarks | RegisterMarks |
| CIP3BeginColorControl<br>CIP3EndColorControl | ColorControl | ColorControl |
| CIP3BeginCutData<br>CIP3EndCutData | CutData | CutData |
| CIP3BeginCutBlock<br>CIP3EndCutBlock | CutBlock | *inherited from the CIP3BlockName attribute specified inside the structure* |
| CIP3BeginFoldProcedures<br>CIP3EndFoldProcedures | FoldProcedures | FoldProcedures |
| *name* CIP3BeginPrivate<br>CIP3EndPrivate | PrivateData | *name* |

**Table 3-3: Structure types and names**

**Definition of a valid structure tree in a CIP3 file**

The rules shown below (in a Backus-Naur like notation) specify how to build a valid logical structure for a CIP3 file.

```
Sheet           ::=  Surface?, Surface?, PrivateData*.
Surface         ::=  PreviewImage?, RegisterMarks?, ColorControl?,
                     CutData?, FoldProcedures?, PrivateData*.
PreviewImage    ::=  Separation*, PrivateData*.
CutData         ::=  CutBlock*, PrivateData*.
CutBlock        ::=  CutBlock*, PrivateData*.
PrivateData     ::=  PrivateData*.

Notation:
item            : item must occur exactly once
item?           : item is optional
item+           : item can occur once or more
item*           : item can occur an arbitrary number of times or not at all
item-a, item-b  : list of item-a and item-b (in arbitrary sequence)
```

**Table 3-4: Rules defining all valid CIP3 PPF structure trees**

It is not allowed to have more than one *Front* and one *Back* structure within a *Sheet* structure.

**Example 3-9: Definition of Logical Structure**

This example matches Illustration 2-1 on page 7.

```
%!PS-Adobe-3.0
%%CIP3-File Version 3.0
%ÔÒ¤Ë

   < ... the PPF Directory would be inserted here ... >

   < ... the Product Definition would be inserted here ... >

CIP3BeginSheet
   < ... attribute definitions for both front and back ... >
CIP3BeginFront
   < ... attribute definitions for front ... >
CIP3BeginPreviewImage
   < ... composite (CMYK) preview image of front ... >
CIP3EndPreviewImage
CIP3BeginRegisterMarks
   < ... placing of register marks onto front ... >
/FhGPrivate CIP3BeginPrivate
   < ... private data inside RegisterMarks structure ... >
CIP3EndPrivate
CIP3EndRegisterMarks
CIP3BeginColorControl
   < ... placing of color control fields onto front ... >
CIP3EndColorControl
CIP3BeginCutData
CIP3BeginCutBlock
   < ... cut block 1 ... >
CIP3EndCutBlock
CIP3BeginCutBlock
   < ... cut block 2 ... >
CIP3EndCutBlock
CIP3BeginCutBlock
   < ... cut block 3, containing two nested cut blocks ... >
CIP3BeginCutBlock
   < ... cut block 4 ... >
CIP3EndCutBlock
CIP3BeginCutBlock
   < ... cut block 5 ... >
CIP3EndCutBlock
CIP3EndCutBlock
CIP3EndCutData
CIP3EndFront
CIP3BeginBack
CIP3BeginPreviewImage
   < ... preview image of back with two separations ... >
CIP3BeginSeparation
```

```
   < ... 1st separation of back preview image... >
CIP3EndSeparation
CIP3BeginSeparation
   < ... 2nd separation of back preview image... >
CIP3EndSeparation
CIP3EndPreviewImage
CIP3BeginRegisterMarks
   < ... placing of register marks onto back ... >
CIP3EndRegisterMarks
CIP3EndBack
CIP3EndSheet

   < ... the definition of more sheets would be inserted here ... >

%%CIP3EndOfFile
```

### 3.1.5. What is a Valid CIP3 PPF File?

A valid CIP3 PPF file must meet the following requirements:

1. It must be syntactically correct, i.e. file structure and syntax must meet the specification.
2. The logical structure, i.e. the nesting of structures, must meet the specification.
3. In each structure, all required attributes must be defined. Due to the inheritance mechanism it is allowed to specify an attribute at a higher level.
4. Private attributes may be placed in any structure or dictionary, if they do not conflict with specified dict entries and if they do not use the CIP3 prefix.
5. The data type for the attribute must meet the specification. If an attribute contains a dictionary, all required entries must be defined.
6. CIP3 content may only occur in the appropriate structures listed below. Only CIP3 comments, CIP3 annotations and CIP3 Private Content may appear within any structure.
   - preview image data in PreviewImage and Separation structures
   - register marks in RegisterMarks structure
   - color or density measuring fields and color control strips in ColorControl structure
   - cut marks in CutData and CutBlock structures

## 3.2. PPF Directory

The ***PPFDirectory*** structure contains directory information about all single sheet PPF subfiles contained in the whole CIP3 PPF file. For each PPF subfile a ***CIP3PPFDirEntry*** command is required. It is possible to reserve directory entries for future adding of sheets by setting offset and length to zero. In order to allow for modifications of these entries the PPFDirectory structure uses a fixed length format with 256 bytes for each entry.

The PPFDirectory is required if either more than one sheet is encoded in the file or a Product definition section is there. Otherwise it is optional (see Illustration 3-1: “Syntactical structure of the CIP3 file”).

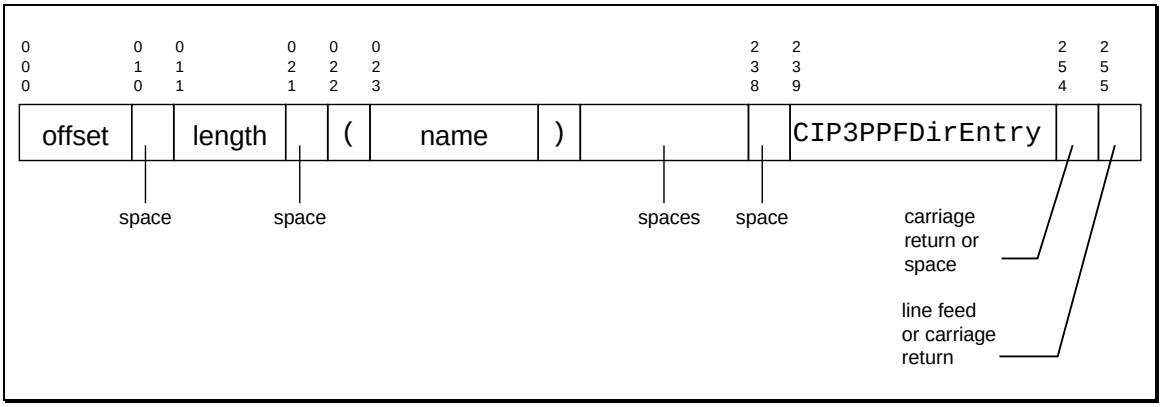



**Illustration 3-3: Fixed size directory entry**

| Parameter | Offset | Length | Type | Necessary | Meaning |
|---|---|---|---|---|---|
| ***offset*** | 0 | 10 | integer | required | Address of the first byte of the single sheet PPF subfile within the complete CIP3 PPF file. A value of zero indicates a missing sheet.<br>The number must be coded in 10 byte fixed format using leading blanks or zeros. |
| ***length*** | 11 | 10 | integer | required | Length in bytes of the single sheet PPF subfile. If the subfile is missing, the value zero must be used.<br>The number must be coded in 10 byte fixed format using leading blanks or zeros. |
| ***name*** | 22 | 216 | string | required | Sheet name. This name must be copied from the CIP3AdmSheetName attribute as defined in the PPF subfile. The name must be a unique sheet name (at least within the whole CIP3 PPF file). It must not contain a slash character (or the Unicode equivalent character, if coded in Unicode).<br>The string must be coded in up to 216 bytes (including the parentheses needed to build a PostScript string, even if it is coded in Unicode). |

**Table 3-5: Parameter for the CIP3PPFDirEntry command**

**Example 3-10: PPF Directory of a CIP3 PPF Describing a Brochure with 48 Pages**

Please note that in the following example word "CIP3PPFDirEntry" starts in column 239 (assuming the first column is called column 0).

```
CIP3BeginPPFDirectory
0000003116 0000655193 (Cover = pages 1,2,43,44)                    CIP3PPFDirEntry
0000658309 0000120080 (Pages 3,4,41,42)                            CIP3PPFDirEntry
0000000000 0000000000 (Pages 5,6,39,40)                            CIP3PPFDirEntry
0000000000 0000000000 (Pages 7,8,37,38)                            CIP3PPFDirEntry
0000000000 0000000000 (Pages 9,10,35,36)                           CIP3PPFDirEntry
0000000000 0000000000 (Pages 11,12,33,34)                          CIP3PPFDirEntry
0000000000 0000000000 (Pages 13,14,31,32)                          CIP3PPFDirEntry
0000000000 0000000000 (Pages 15,16,29,30)                          CIP3PPFDirEntry
0000000000 0000000000 (Pages 17,18,27,28)                          CIP3PPFDirEntry
0000000000 0000000000 (Pages 19,20,25,26)                          CIP3PPFDirEntry
0000000000 0000000000 (Pages 21,22,23,24)                          CIP3PPFDirEntry
0000000000 0000000000 (Insert = without page numbers)              CIP3PPFDirEntry
CIP3EndPPFDirectory
```

## 3.3. Product Definition

Within the CIP3 Print Production Format the production of a complete product, e.g. a brochure, can be specified in a section called Product structure. Like all structures in the CIP3 PPF, it is enclosed in appropriate begin and end structure commands: ***CIP3BeginProductDefinition*** and ***CIP3EndProductDefinition***.

It is possible to describe more than one product within the Product structure. Complex products may be specified by a series of partial products. In this case a composite product results from executing an operation which uses other products as input components. These input components are then called "partial products". They must be defined in the same file.

The product definition section is optional. If it is present, it must follow the PPF Directory and it must occur before the first sheet. It is not allowed to have more than one product definition section in a CIP3 PPF file. Nesting of product definitions is not allowed.

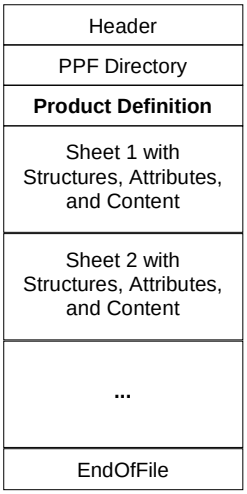



**Illustration 3-4: Position of the product definition in a CIP3 PPF file**

There are only two attributes required in the Product structure:

- The ***CIP3Products*** attribute contains an array of Product Definition dictionaries each describing one product definition step.
- The ***CIP3FinalProducts*** attribute is an array specifying one or more complete products. Each product name contained in this array must match with the CIP3ProductName attribute defined in one of the Product Definition dictionaries. Each of the products listed in the CIP3FinalProducts array acts as the root of the definition of one product.

A product is defined by a sequence of product definition steps. Each step is defined by specifying a dictionary containing the parameters of the product definition step. All these product definition dictionaries must be contained in the *CIP3Products* array. There is no meaning imposed by the sequence, in which the dictionaries occur in this array.

| Attribute Name | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3Products** | array of dictionaries | required | Array containing all Product Definition dictionaries (see Table 3-7: “Product Definition dictionary”) |
| **CIP3FinalProducts** | array of strings | required | Names of products (not including the partial products) |

**Table 3-6: Attributes of Product Definition structure**

**Example 3-11: Product Definition Structure**

```
CIP3BeginProductDefinition
/CIP3Products [
<<        < ... product definition step 1 ...>
>>
<<        < ... product definition step 2 ...>
>>
] def
/CIP3FinalProducts [ (name of final product) ] def
CIP3EndProductDefinition
```

## 3.3.1. Product Definition Step

### 3.3.1.1. Overview

Each product definition step describes one of the operations that are necessary to build the desired product. It is defined by specifying a Product Definition dictionary (see Table 3-7: ”Product Definition dictionary”) containing the following information:

- the **name of the (partial) product**
- the **type of the operation**
  e.g. AdhesiveBinding (see Table 3-10: “Operation types” for a complete list of operation types)
- some **operation specific parameters**.
  Parameters that are valid for the whole operation (e.g. the type of glue to be used) are stored in an *operation specific dictionary*. Since the content of this dictionary depends on the type of the operation, it’s possible entries are specified separately for each operation.
- the **input components** used by this operation (e.g. a set of folded sheets)
  The input components are defined by specifying an array of dictionaries (see Table 3-8: “Component definition dictionary”) each specifying one input component. The order in which the components appear in the dictionary is important. Parameters that are specific to a single input component of the operation can be defined in a *component specific dictionary*. Since the content of this dictionary depends on the type of the operation, it’s possible entries are specified separately for each operation.
- some **product related administration information**
  Especially for final products some additional attributes can be defined, like job name, customer name, or copyright.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/CIP3ProductName** | string | required | Name of product.<br>At least within one CIP3 PPF file, product names must be unique. |
| **/CIP3ProductOperation** | name | required | Type of operation (see chapter 3.3.2) |
| **/CIP3ProductParams** | dictionary | optional | Dictionary containing operation specific attributes. The elements of this dictionary are specified in the description of each operation (see chapter 3.3.2) |
| **/CIP3ProductComponents** | array of dictionaries | required | Each dictionary describes one input component of this operation (see Table 3-8: ”Component definition dictionary”). |
| **/CIP3ProductJobName** | string | *see meaning* | Job name of product.<br>This entry is required, if the product is a final product (i.e., if it is listed in the CIP3FinalProducts array; see Table 3-6: ”Attributes of Product Definition structure”) |
| **/CIP3ProductJobCode** | string | optional | Code used for job identification of product |
| **/CIP3ProductCopyright** | string | optional | Copyright |
| **/CIP3ProductCustomer** | string | optional | Name of target product customer |
| **/CIP3ProductVolume** | integer | optional | Number of products to be produced |

**Table 3-7: Product Definition dictionary**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/SourceType** | name | required | Specifies the source type of the component: **/Sheet**, **/Block**, **/PartialProduct**, **/ExternalProduct**.<br>See chapter 3.3.1.2 for more detail about the different source types. |
| **/SourceSheet** | string | *see meaning* | This entry is only required, if */SourceType* is either **/Sheet** or **/Block**.<br>It contains the name of the sheet that should be used as input component of the operation. |
| **/SourceBlock** | string | *see meaning* | This entry is only required, if */SourceType* = **/Block**.<br>It contains the structure path of name of the sheet that should be used as input component of the operation.<br>See chapter 3.3.1.2 for more detail. |
| **/SourceProduct** | string | *see meaning* | This entry is only required, if */SourceType* = **/PartialProduct**.<br>It contains the name of the partial product. |
| **/Params** | dictionary | optional | Dictionary containing the component specific parameters of an operation (see chapter 3.3.2 for more detail). |

**Table 3-8: Component definition dictionary**

#### 3.3.1.2. Referencing Input Components

As described in the overview chapter 3.3.1.1 the input components of an operation can be of one of four source types. Since some of them require the definition of specific entries in the component definition dictionary, they have to be described in more detail:

**/Sheet** This source type is appropriate, if a flat sheet (e.g. a postcard to be glued in) is used as an input component. "Flat" in this case means, that the sheet has not been folded or cut before the operation.
The sheet name - as it appears in the PPF directory - must be specified in the *SourceSheet* entry of the component definition dictionary.

**/Block** This source type is appropriate, if a folded sheet , a cut portion of the sheet, or a cut and folded portion of a sheet is used as an input component. Please note that a dummy cut block (*CIP3BlockType* = **/TempBlock**) covering the whole area of the sheet must be defined in order to use a folded sheet as an input component.
The sheet name - as it appears in the PPF directory - must be specified in the *SourceSheet* entry of the component definition dictionary.
In addition, the structure path of the requested cut block must be specified in the *SourceBlock* entry of the component definition dictionary. This is done by building a PostScript string containing all structure names starting from the Sheet structure up to the requested cut block structure. Within that string each structure name is preceded by a slash character ('/').
Example: (/Sheet/Front/CutData/Block 1).

**/PartialProduct** This source type is appropriate, if a partial product should be used as an input component.
The product name - as specified in the corresponding product definition step - must be specified in the *SourceProduct* entry of the component definition dictionary.

**/ExternalProduct** This source type is appropriate, if a product should be used as an input component, which is not defined within the same CIP3 PPF file. This mechanism can only be used, if no further information about the external product is required to perform the operation.

**Example 3-12: Product Definition**

This example shows the definition of a small 8-page booklet. It is built from two folded 4- page sheets, which are collected and then stitched together.

```
CIP3BeginProductDefinition
/CIP3Products [
<<
     /CIP3ProductName (collected sheets)   % this is a partial product
     /CIP3ProductOperation /Collecting
     /CIP3ProductComponents
     [
       <<   % first component
            /SourceType /Block
            /SourceSheet (pages 3,4,5,6)
            /SourceBlock (/Sheet/Front/CutData/Block 1)
            /Params << /Orientation [1 0 0 1 0 0] >>
       >>
       <<   % second component
            /SourceType /Block
            /SourceSheet (pages 1,2,7,8)
            /SourceBlock (/Sheet/Front/CutData/Block 1)
            /Params << /Orientation [1 0 0 1 0 0] >>
       >>
     ]
>>

<<
     /CIP3ProductName (stitched booklet)    % this is a root product
     /CIP3ProductOperation /SaddleStitching
     /CIP3ProductParams   <<
                             /NumberOfStitches 2
                             /StitchPositions [ 105 mm 210 mm ]
                             /StapleShape /Eyelet
                          >>
     /CIP3ProductComponents
     [
       <<   % needs only one component
            /SourceType /PartialProduct
            /SourceProduct (collected sheets)
            /Params << /Orientation [1 0 0 1 0 0] >>
       >>
     ]
>>
] def

/CIP3FinalProducts [ (stitched booklet) ] def
CIP3EndProductDefinition
```

**Example 3-13: Product Definition Step Using a Named Dictionary**

If more than one component needs the same definitions in the component specific dictionary, it is useful to define a named dictionary and reference that dictionary by writing the name without the leading slash character. In this example all components have the same orientation. Therefore a dictionary named “StdOrientation” is defined.

```
CIP3BeginProductDefinition
% define a named dictionary, which can be used later
/StdOrientation << /Orientation [1 0 0 1 0 0] >> def

/CIP3Products [
<<
  /CIP3ProductName (gathered sheets)
  /CIP3ProductOperation /Gathering
  /CIP3ProductComponents
  [
   << /SourceType /Sheet /SourceSheet (pages 1,2) /Params StdOrientation >>
   << /SourceType /Sheet /SourceSheet (pages 3,4) /Params StdOrientation >>
   << /SourceType /Sheet /SourceSheet (pages 5,6) /Params StdOrientation >>
   << /SourceType /Sheet /SourceSheet (pages 7,8) /Params StdOrientation >>
  ]
>>
] def
/CIP3FinalProducts [ (gathered sheets) ] def
CIP3EndProductDefinition
```

#### 3.3.1.3. Coordinate Systems

Each input component of an operation has by default its own coordinate system, which is called source or component coordinate system. This coordinate system also implies a specific orientation of that component.

On the other hand there is a coordinate system, which is used for the specification of some operation specific parameters. This coordinate system is called target or operation coordinate system.

It is often necessary to change the orientation of an input component before executing the operation. This can be done by specifying a PostScript transformation matrix. It is stored in the *Orientation* entry of the component specific dictionary. This allows to specify different matrices for the individual input components of an operation.

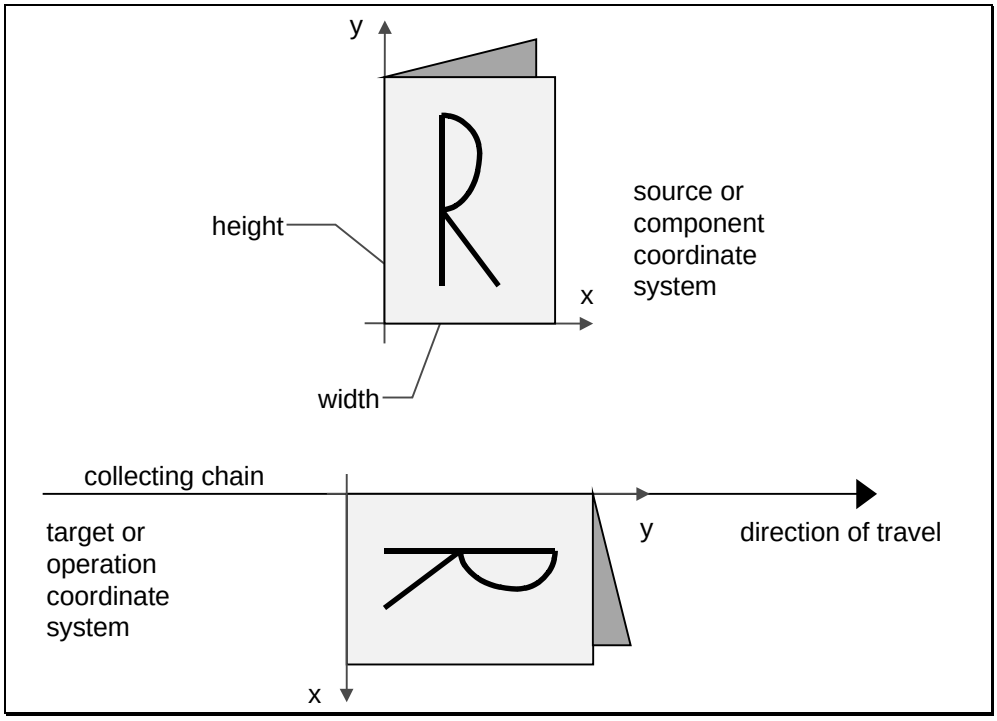



**Illustration 3-5: Coordinate systems used for collecting**

The following table shows some matrices that can be used to change the orientation of an input component.

Please note, that most of them require the knowledge of the width (***w***) and/or the height (***h***) of the input component.

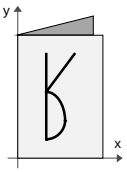
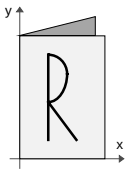
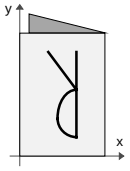
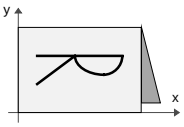
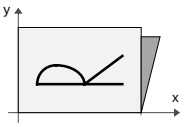
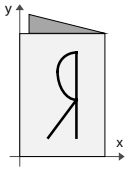
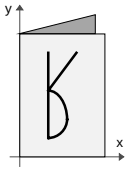
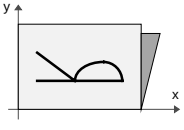
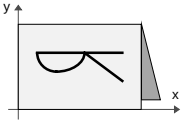
| target coordinate system / source coordinate system |  |  |
|---|---|---|
|  | [ 1 0 0 1 0 0 ] | [ 1 0 0 -1 0 ***h*** ] |
|  | [ -1 0 0 -1 ***w*** ***h*** ] | [ -1 0 0 1 ***w*** 0 ] |
|  | [ 0 1 -1 0 ***w*** 0 ] | [ 0 -1 -1 0 ***w*** ***h*** ] |
|  | [ 0 -1 1 0 0 ***h*** ] | [ 0 1 1 0 0 0 ] |
|  | [ -1 0 0 1 ***w*** 0 ] | [ -1 0 0 -1 ***w*** ***h*** ] |
|  | [ 1 0 0 -1 0 ***h*** ] | [ 1 0 0 1 0 0 ] |
|  | [ 0 1 1 0 0 0 ] | [ 0 -1 1 0 0 ***h*** ] |
|  | [ 0 -1 -1 0 ***w*** ***h*** ] | [ 0 1 -1 0 ***w*** 0 ] |

**Table 3-9: Matrices used to change the orientation**

### 3.3.1.4. Terms and Definitions

The examples of this specification are using some terms, whose meaning depends on the culture in which they are used (e.g. the front side of a magazine). There are other terms, which are defined by the production process and therefore do not depend on the culture (e.g. binding edge).

If possible, we are trying to use the culture independent terms within this specification. In case where this is not possible Western style (i.e. left to right writing) is assumed. Please note that these terms may have a different meaning in other cultures (e.g. in those writing from right to left).

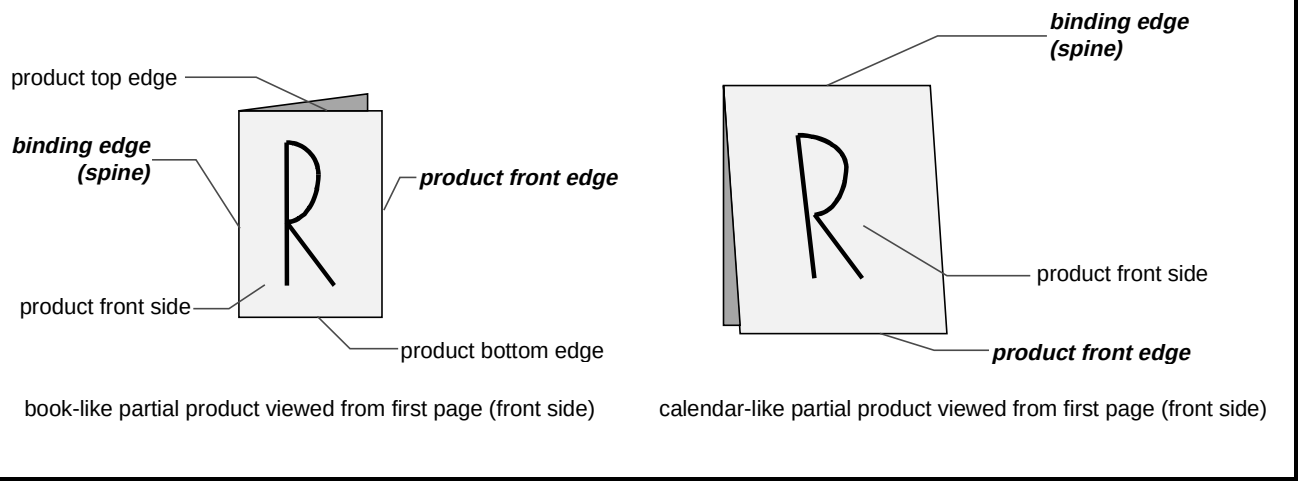



**Illustration 3-6: Folded sheet**

**Definitions:**

binding edge : the edge on which the (partial) product is glued or stitched. This edge is also often called *working edge* or *spine*.

product front edge : the side, where you open the (partial) product. This edge is opposite to the binding edge.

registered edge : a side on which a collection of sheets or partial products is aligned during a production step. All production steps require two registered edges, which must not be opposite to each other. The two registered edges define the coordinate system used within the production step. When there is a binding edge, this is one of the registered edges.

### 3.3.2. Operation Types

The following table lists the operation types which are currently defined:

| Operation Name | Meaning |
|---|---|
| **/Collecting** | Collect sheets on a saddle |
| **/Gathering** | Gathering sheets on a pile |
| **/ThreadSewing** | Thread sewing |
| **/SaddleStitching** | Saddle stitching |
| **/Stitching** | Stitching |
| **/SideSewing** | Side sewing |
| **/EndSheetGluing** | End sheet gluing |
| **/AdhesiveBinding** | Adhesive Binding |
| **/Trimming** | Trimming |
| **/GluingIn** | Gluing in |
| **/Folding** | Folding |

**Table 3-10: Operation types**

In the future there will be more operations available, e.g. like imprinting, labeling, inserting, laminating, embossing, packaging, and palletizing.

In the following sections each operation is specified in more detail.

### 3.3.2.1. Collect On A Saddle

This operation describes the collection of folded sheets, which might have been cut, on a saddle. The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/Collecting**.

The sequence of the collected sheets is specified by the order in which the components appear in the sequence of **CIP3ProductComponents** array: The first component lies at the bottom of the pile collected on the saddle.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component (see Table 3-9: “Matrices used to change the orientation”). |

**Table 3-11: Component specific dictionary for Collecting operation**

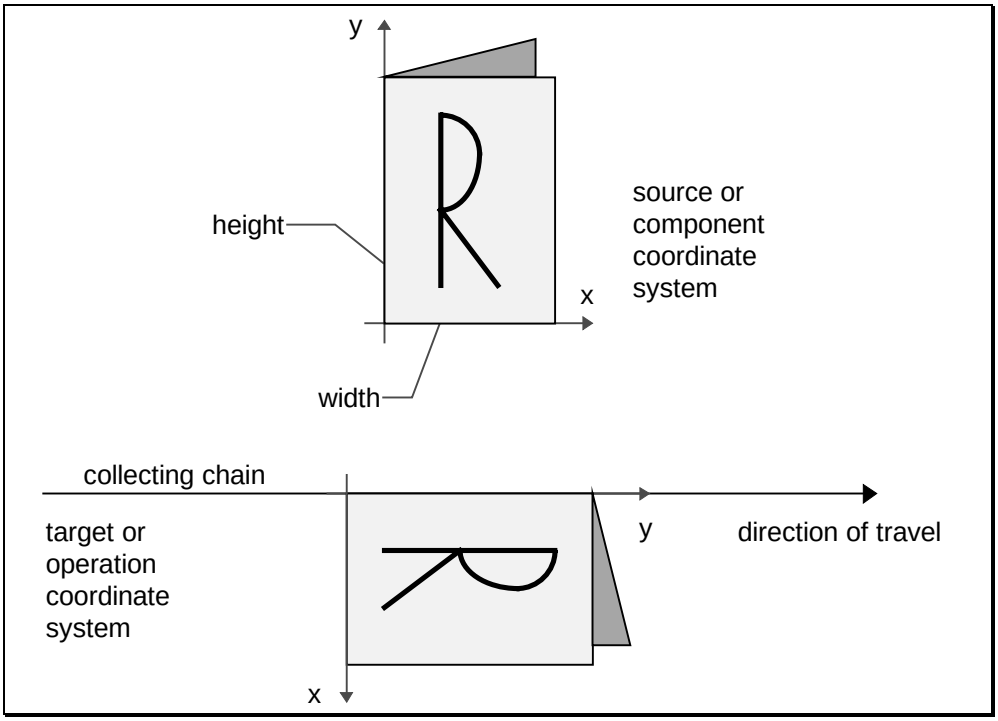



**Illustration 3-7: Coordinate systems used for collecting**

The operation coordinate system is defined as follows:
The y-axis is aligned with the binding edge. It increases from the registered edge to the edge opposite to the registered edge. The x-axis is aligned with the registered edge. It increases from the binding edge to the edge opposite to the binding edge (i.e. the product front edge).

**Example 3-14: Collect On A Saddle**

```
CIP3BeginProductDefinition
/CIP3Products [
<<
     /CIP3ProductName (collected sheets)
     /CIP3ProductOperation /Collecting
     /CIP3ProductComponents
     [
       <<   % first component
            /SourceType /Block
            /SourceSheet (pages 3,4,5,6)
            /SourceBlock (/Sheet/Front/CutData/Block 1)
            /Params << /Orientation [1 0 0 1 0 0] >>
       >>
       <<   % second component
            /SourceType /Block
            /SourceSheet (pages 1,2,7,8)
            /SourceBlock (/Sheet/Front/CutData/Block 1)
            /Params << /Orientation [1 0 0 1 0 0] >>
       >>
     ]
>>
] def

/CIP3FinalProducts [ (collected sheets) ] def
CIP3EndProductDefinition
```

### 3.3.2.2. Gathering On A Pile

This operation describes the gathering of sheets or partial products on a pile. The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/Gathering**.

The sequence of the gathered sheets is specified by the order in which the components appear in the sequence of *CIP3ProductComponents* array: The first component lies at the bottom of gathering channel.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component. |

**Table 3-12: Component specific dictionary for Gathering operation**

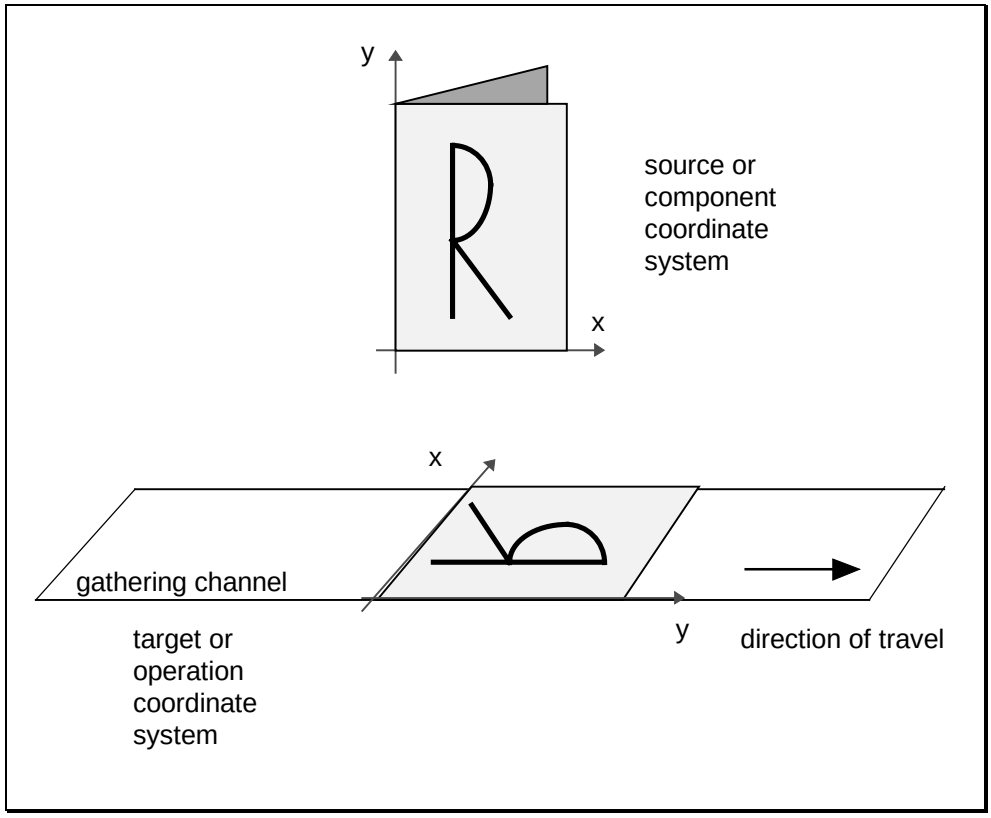



**Illustration 3-8: Coordinate systems used for gathering**

The operation coordinate system is defined as follows:
The y-axis is aligned with the binding edge. It increases from the registered edge to the edge opposite to the registered edge. The x-axis is aligned with the registered edge. It increases from the binding edge to the edge opposite to the binding edge (i.e. the product front edge).

**Example 3-15: Gathering On A Pile**

```
CIP3BeginProductDefinition
/CIP3Products [
<<
   /CIP3ProductName (gathered sheets)
   /CIP3ProductOperation /Gathering
   /CIP3ProductComponents
   [
      <<    /SourceType /Sheet
            /SourceSheet (pages 1,2)
            /Params << /Orientation [1 0 0 1 0 0] >>
      >>
      <<    /SourceType /Sheet
            /SourceSheet (pages 3,4)
            /Params << /Orientation [1 0 0 1 0 0] >>
      >>
      <<    /SourceType /Sheet
            /SourceSheet (pages 5,6)
            /Params << /Orientation [1 0 0 1 0 0] >>
      >>
      <<    /SourceType /Sheet
            /SourceSheet (pages 7,8)
            /Params << /Orientation [1 0 0 1 0 0] >>
      >>
   ]
>>
] def

/CIP3FinalProducts [ (gathered sheets) ] def
CIP3EndProductDefinition
```

### 3.3.2.3. Thread Sewing

This operation describes the thread sewing process. The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/ThreadSewing**.

This operation may include a gluing application, which is mostly being used between the first and the second or the last and the last but one sheet. In addition gluing may be necessary if different types of paper are used.

The operation requires one component, the gathered sheets.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/NumberOfNeedles** | integer | required | Number of needles. |
| **/NeedlePositions** | array of numbers | optional | Array containing the y-coordinate of the needle positions. The number of array elements must match the number given in NumberOfNeedles. |
| **/CoreMaterial** | name | optional | Core material of thread: **/Cotton**, **/Nylon**, **/Polyester.**<br>This attribute must be used to define the thread material, if there is no casting. |
| **/CastingMaterial** | name | optional | Casting material of thread: **/Cotton**, **/Nylon**, **/Polyester.** |
| **/ThreadThickness** | number | optional | Thread thickness. |
| **/ThreadBrand** | string | optional | Thread brand. |
| **/SewingPattern** | name | optional | Sewing pattern: **/Normal**, **/Staggered**, **/CombinedStaggered** |
| **/GlueLine** | dictionary | optional | Gluing dictionary (see Table 3-9: “Matrices used to change the orientation”). |
| **/GlueLineRefSheets** | array of integers | *see meaning* | This dictionary entry is only required, if **/GlueLine** is defined.<br>It contains the indices of components (of the predecessing gathering operation) after which liming should be applied. The index starts with 0. |
| **/BlindStitch** | boolean | required | True, if blind stitch after last stitch is required. |
| **/Sealing** | boolean | required | True, if thermo-sealing is required. |

**Table 3-13: Operation specific dictionary for ThreadSewing operation**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component.<br>The binding edge of the book block (spine) is assumed to be at the Y-axis after applying this matrix to the book block. |

**Table 3-14: Component specific dictionary for ThreadSewing operation**

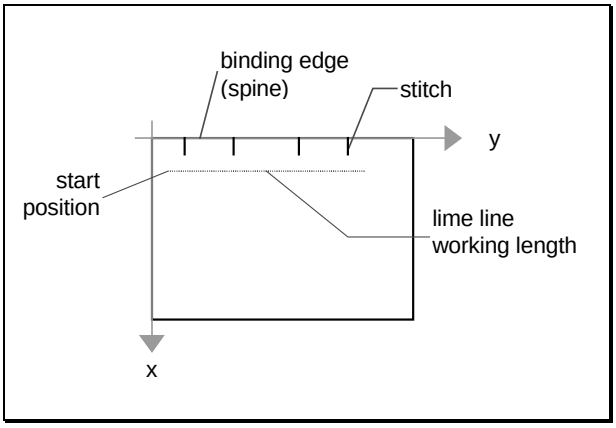



**Illustration 3-9: Parameters and coordinate system used for thread sewing**

The operation coordinate system is defined as follows:
The y-axis is aligned with the binding edge. It increases from the registered edge to the edge opposite to the registered edge. The x-axis is aligned with the registered edge. It increases from the binding edge to the edge opposite to the binding edge (i.e. the product front edge).

**Example 3-16: Thread Sewing**

```
CIP3BeginProductDefinition
/CIP3Products [
<<
   /CIP3ProductName (sewed book block)
   /CIP3ProductOperation /ThreadSewing
   /CIP3ProductParams <<
                           /NumberOfNeedles 4
                           /GlueLineRefSheets [ 0 ]
                           /GlueLine <<
                                        /StartPosition [ 15 mm 10 mm ]
                                        /WorkingPath [ 15 mm 287 mm ]
                                        /GluingPattern []
                                     >>
                           /BlindStitch false
                           /Sealing false
                       >>
   /CIP3ProductComponents
   [
       <<   % book block before sewing
            /SourceType /PartialProduct
            /SourceProduct (book block)
            /Params <</Orientation [1 0 0 1 0 0]>>
       >>
   ]
>>

<<
   /CIP3ProductName (book block)
   % ... the definition of the book block operation would go here ...
>>
] def

/CIP3FinalProducts [ (sewed book block) ] def
CIP3EndProductDefinition
```

### 3.3.2.4. Saddle Stitching

This operation describes the saddle stitching process. The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/SaddleStitching**.

The operation requires one component, the collected sheets.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/NumberOfStitches** | integer | required | Number of stitches |
| **/StitchPositions** | array of numbers | optional | Array containing the stitch positions. The center of the stitch must be specified (see Illustration 3-11: “Parameters and coordinate system used for saddle stitching”). The number of array elements must match the number given in NumberOfStitches |
| **/StapleShape** | name | optional | Shape of staples: **/Crown**, **/Overlap**, **/Butted, /ClinchOut**, **/Eyelet** (see Illustration 3-10: “Staple shapes”). |
| **/StitchWidth** | number | optional | Width of stitch. |
| **/WireGauge** | number | optional | Wire width |
| **/WireBrand** | string | optional | Wire brand |

**Table 3-15: Operation specific dictionary for SaddleStitching operation**

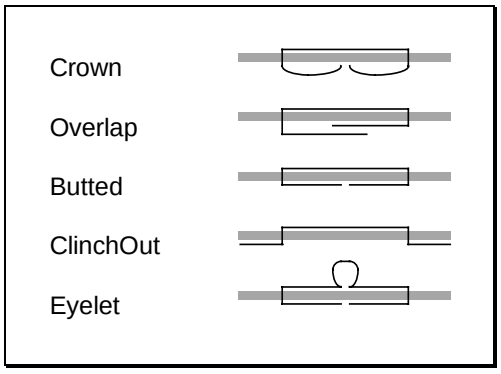



**Illustration 3-10: Staple shapes**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component.<br>The binding edge of the book block (spine) is assumed to be at the Y-axis after applying this matrix to the book block. |

**Table 3-16: Component specific dictionary for SaddleStitching operation**

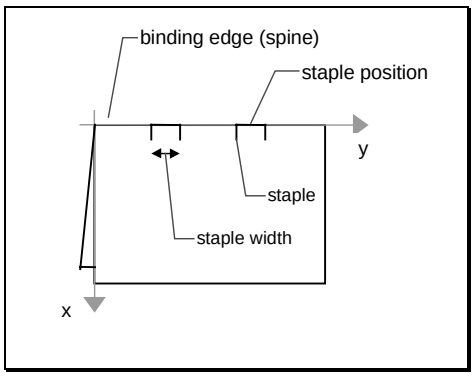



**Illustration 3-11: Parameters and coordinate system used for saddle stitching**

The operation coordinate system is defined as follows:
The y-axis is aligned with the binding edge. It increases from the registered edge to the edge opposite to the registered edge. The x-axis is aligned with the registered edge. It increases from the binding edge to the edge opposite to the binding edge (i.e. the product front edge).

**Example 3-17: Saddle Stitching**

```
CIP3BeginProductDefinition
/CIP3Products [
<<
   /CIP3ProductName (saddle stitched booklet)
   /CIP3ProductOperation /SaddleStitching
   /CIP3ProductParams <<
                          /NumberOfStitches 2
                          /StitchPositions [105 mm 210 mm]
                          /StapleShape /Crown
                          /StitchWidth 12 mm
                       >>
   /CIP3ProductComponents
   [
      <<
            % collected sheets before saddle stitching
            /SourceType /PartialProduct
            /SourceProduct (collected sheets)
            /Params <</Orientation [1 0 0 1 0 0]>>
      >>
   ]
>>

<<
   /CIP3ProductName (collected sheets)
   % ... the definition of the collecting operation would go here ...
>>
] def

/CIP3FinalProducts [(saddle stitched booklet) ] def
CIP3EndProductDefinition
```

### 3.3.2.5. Stitching

This operation describes the side stitching process. The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/Stitching**.

The operation requires one component, the gathered or collected sheets.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/NumberOfStitches** | integer | required | Number of stitches. |
| **/StitchPositions** | array of numbers | optional | Array containing the stitch positions. The center of each stitch must be specified (see Illustration 3-12: “Parameters and coordinate system used for stitching”). The number of array elements must match the number given in NumberOfStitches. |
| **/StitchWidth** | number | optional | Width of stitch. |
| **/StapleShape** | name | optional | Shape of staples: **/Crown**, **/Overlap**, **/Butted, /ClinchOut**, **/Eyelet** (see Illustration 3-10: “Staple shapes”). |
| **/Offset** | number | required | Distance between stitch and binding edge. |
| **/Angle** | number | optional | Angle of stitch in degree. The angle is given counterclockwise.<br>0 = horizontal (i.e. parallel to the X axis of the operation coordinate system) |
| **/StitchFromFront** | boolean | required | If true, stitching is done from front to back. Otherwise it is done from back to front. |
| **/WireGauge** | number | optional | Wire width in mm |
| **/WireBrand** | string | optional | Wire brand |

**Table 3-17: Operation specific dictionary for Stitching operation**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component.<br>The binding edge is assumed to be at the Y-axis after applying this matrix to the book block. |

**Table 3-18: Component specific dictionary for Stitching operation**

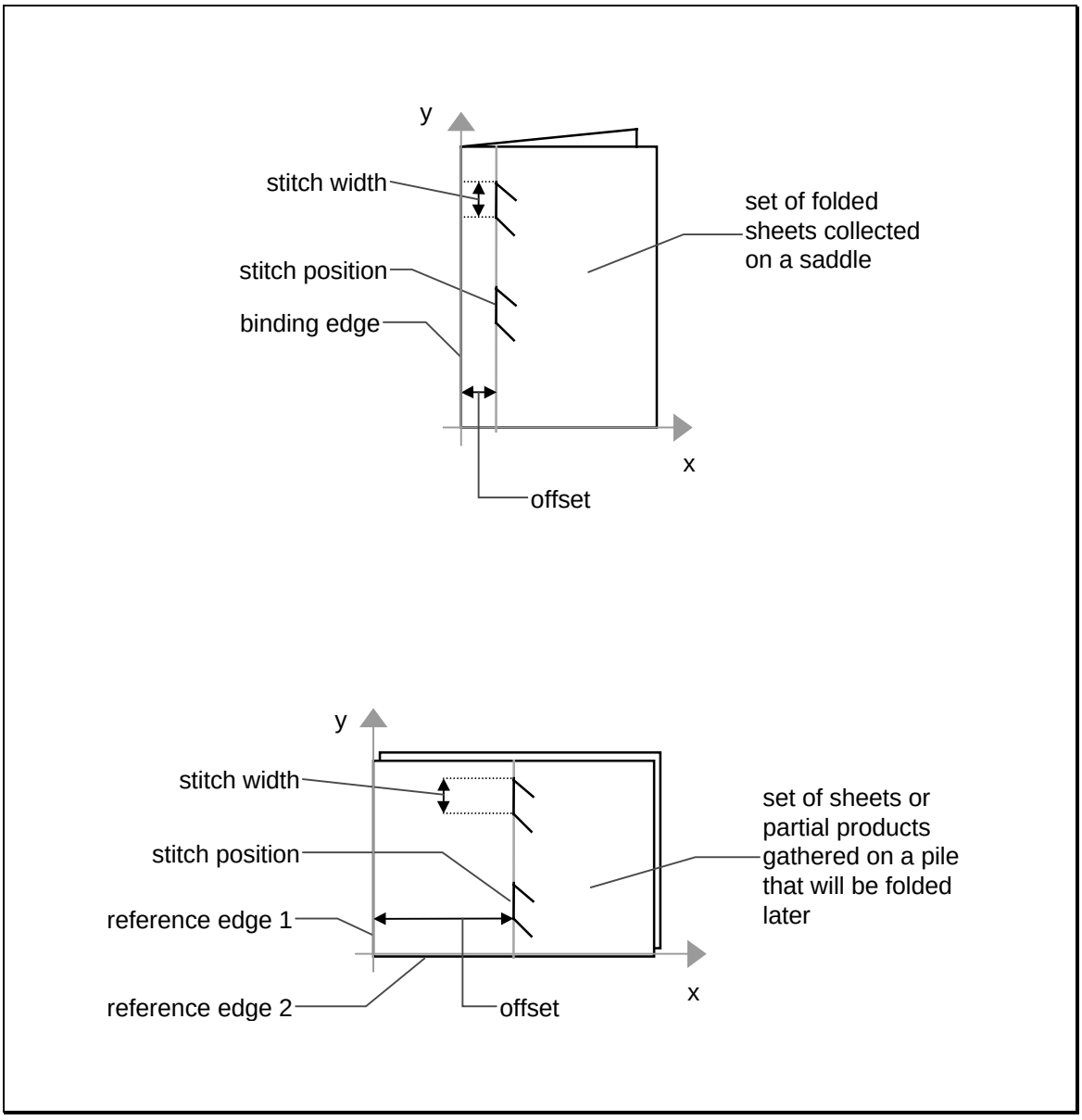



**Illustration 3-12: Parameters and coordinate system used for stitching**

**Example 3-18: Stitching**

```
CIP3BeginProductDefinition
/CIP3Products [
<<
   /CIP3ProductName (stitched booklet)
   /CIP3ProductOperation /Stitching
   /CIP3ProductParams  <<
                           /NumberOfStitches 2
                           /StitchPositions [105 mm 210 mm]
                           /StapleShape /Butted
                           /StitchWidth 12 mm
                           /Angle 90
                           /Offset 12 mm
                           /StitchFromFront true
                        >>
   /CIP3ProductComponents
   [
      << % collected sheets before saddle stitching
         /SourceType /PartialProduct
         /SourceProduct (collected sheets)
         /Params << /Orientation [1 0 0 1 0 0] >>
      >>
   ]
>>

<<
   /CIP3ProductName (collected sheets)
   % ... the definition of the collecting operation would go here ...
>>
] def

/CIP3FinalProducts [ (stitched booklet) ] def
CIP3EndProductDefinition
```

### 3.3.2.6. Side Sewing

This operation describes the side sewing process. The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/SideSewing**.

The operation requires one component, the gathered sheets.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/NumberOfNeedles** | integer | required | Number of needles |
| **/NeedlePositions** | array of numbers | optional | Array containing the y-coordinates of the needle positions. The number of array elements must match the number given in NumberOfNeedles. |
| **/Offset** | number | required | Distance between stitch and binding edge. |
| **/ThreadMaterial** | name | optional | Thread material: **/Cotton**, **/Nylon**, **/Polyester** |
| **/ThreadThickness** | number | optional | Thread thickness. |
| **/ThreadBrand** | string | optional | Thread brand. |
| **/SewingPattern** | name | optional | Sewing pattern: **/Normal**, **/Staggered**, **/CombinedStaggered** |

**Table 3-19: Operation specific dictionary for SideSewing operation**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component.<br>The binding edge is assumed to be at the Y-axis after applying this matrix to the book block. |

**Table 3-20: Component specific dictionary for SideSewing operation**

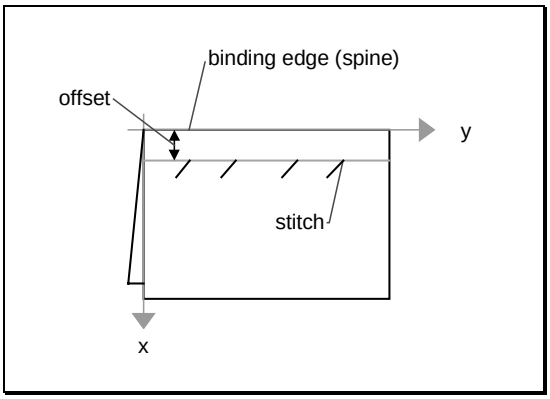



**Illustration 3-13: Parameters and coordinate system used for side sewing**

**Example 3-19: Side Sewing**

```
CIP3BeginProductDefinition
/CIP3Products [
<<
   /CIP3ProductName (side sewed booklet)
   /CIP3ProductOperation /SideSewing
   /CIP3ProductParams <<
                           /NumberOfNeedles 4
                           /NeedlePositions [30 mm 40 mm 260 mm 270 mm]
                           /Offset 12 mm
                           /ThreadMaterial /Nylon
                           /ThreadThickness 0.3 mm
                           /SewingPattern /Normal
                        >>
   /CIP3ProductComponents
   [
      <<
           % collected sheets before side sewing
           /SourceType /PartialProduct
           /SourceProduct (gathered sheets)
           /Params <</Orientation [1 0 0 1 0 0]>>
      >>
   ]
>>

<<
   /CIP3ProductName (gathered sheets)
   % ... the definition of the gathering operation would go here ...
>>
] def

/CIP3FinalProducts [ (side sewed booklet) ] def
CIP3EndProductDefinition
```

### 3.3.2.7. End Sheet Gluing

This operation describes the end sheet gluing process. End sheet gluing is a preparation step for case binding. The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/EndSheetGluing**.

The operation requires three components in the following sequence:

- end sheet (back)
- book block (collected sheets)
- end sheet (front)

The following component specific attribute must only be used for the book block component.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component (see Table 3-9: “Matrices used to change the orientation”). |

**Table 3-21: Component specific dictionary for Collecting operation (for book block component only)**

The following component specific attributes must only be used for the front and the back end sheet components.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component.<br>The binding edge of the book block (spine) is assumed to be at the Y-axis after applying this matrix to the book block. |
| **/Offset** | array of 2 numbers | required | offset of end sheet in X and Y direction. |
| **/GlueLine** | dictionary | required | Gluing dictionary (see Table 3-32: “GlueLine Dictionary describing a Glue Line”). |

**Table 3-22: Component specific dictionary for EndSheetGluing operation (for front and back end sheet components only)**

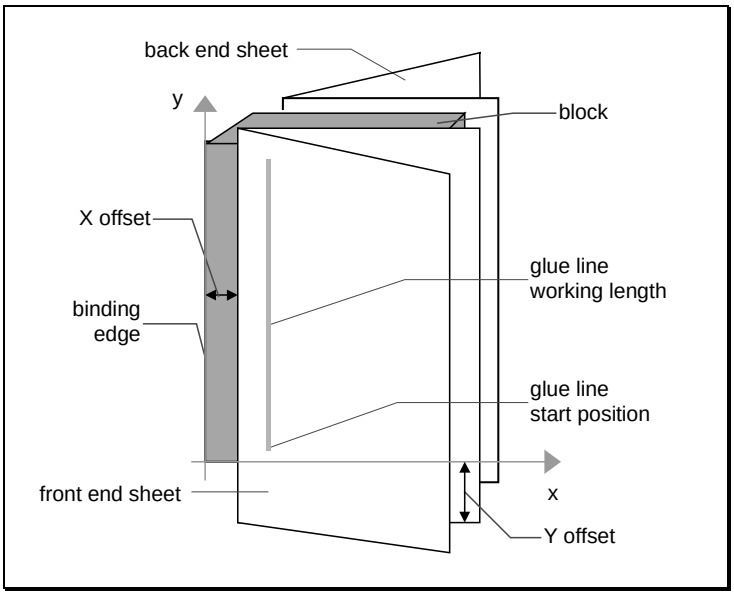



**Illustration 3-14: Parameters and coordinate system used for end sheet gluing**

The operation coordinate system is defined as follows:
The y-axis is aligned with the binding edge of the book block (which is always the second component). It increases from the registered edge to the edge opposite to the registered edge. The x-axis is aligned with the registered edge. It increases from the binding edge to the edge opposite to the binding edge (i.e. the product front edge).

**Example 3-20: End Sheet Gluing**

```
CIP3BeginProductDefinition
/CIP3Products [
<<
   /CIP3ProductName (book block with end sheets)
   /CIP3ProductOperation /EndSheetGluing
   /CIP3ProductComponents
   [
        <<  % back end sheet
            /SourceType /Block
            /SourceSheet (back end sheet)
            /SourceBlock (/Sheet/Front/CutData/Block 1)
            /Params <<  /Orientation [1 0 0 1 0 0]
                        /Offset [ 5 mm -12 mm ]
                        /GlueLine <<
                                    /StartPosition [ 15 mm 3 mm ]
                                    /WorkingPath [15 mm 294 mm ]
                                    /GlueType /ColdGlue
                                    /GluingPattern []
                                    /GlueLineWidth 1 mm
                                  >>
                    >>
```

```
        >>

        <<  % book block
            /SourceType /PartialProduct
            /SourceProduct (book block)
            /Params <<  /Orientation [1 0 0 1 0 0] >>
        >>

        <<  % front end sheet
            /SourceType /Block
            /SourceSheet (front end sheet)
            /SourceBlock (/Sheet/Front/CutData/Block 1)
            /Params <<  /Orientation [1 0 0 1 0 0]
                        /Offset [ 5 mm -12 mm ]
                        /GlueLine <<
                                    /StartPosition [ 15 mm 3 mm ]
                                    /WorkingPath [ 15 mm 294 mm ]
                                    /GluingPattern []
                                    /GlueLineWidth 1 mm
                                >>
                    >>
        >>
   ]
>>

<<
   /CIP3ProductName (book block)
   % ... the definition of the book block operation would go here ...
>>
] def

/CIP3FinalProducts [(book block with end sheets) ] def
CIP3EndProductDefinition
```

### 3.3.2.8. Adhesive Binding

This operation describes the adhesive binding process. It can be used for perfect binding or for book block production and can include the following four subprocesses:

- back preparation
- multiple glue applications
- lining
- cover application

Each subprocess is optional. The parameters for each subprocesses are defined in separate dictionaries, which are used in the Processes entry of the operation specific dictionary for AdhesiveBinding (see Table 3-23: “Operation specific dictionary for AdhesiveBinding operation”).

The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/AdhesiveBinding**.

The operation requires one or two components in the following sequence:

- book block
- cover (optional)

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Processes** | array of dictionaries | required | Each dictionary describes the parameters of one single process of complete AdhesiveBinding operation.<br>The processes should be performed in the same order as they appear in the array of dictionaries.<br>The type of the particular process can be identified by the /ProcessType entry inside the dictionary.<br>Each dictionary must be of one of the following types: BackPreparation (see Table 3-25: “Dictionary for Back Preparation Process”), GlueApplication (see Table 3-26: “Dictionary for Glue Application”), Lining (see Table 3-27: “Dictionary for Lining Process”), or CoverApplication (see Table 3-28: “Dictionary for Cover Applications”). |
| **/PullOutValue** | number | optional | Pull out quality parameter given in [N/cm]. |
| **/PullOutMake** | string | optional | Make of pull out value testing device. |
| **/FlexValue** | number | optional | Flex quality parameter given in [N/cm]. |
| **/FlexMake** | string | optional | Make of flex value testing device. |

**Table 3-23: Operation specific dictionary for AdhesiveBinding operation**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component.<br>The binding edge of the book block (spine) is assumed to be at the Y-axis after applying this matrix to the book block. |

**Table 3-24: Component specific dictionary for AdhesiveBinding operation**

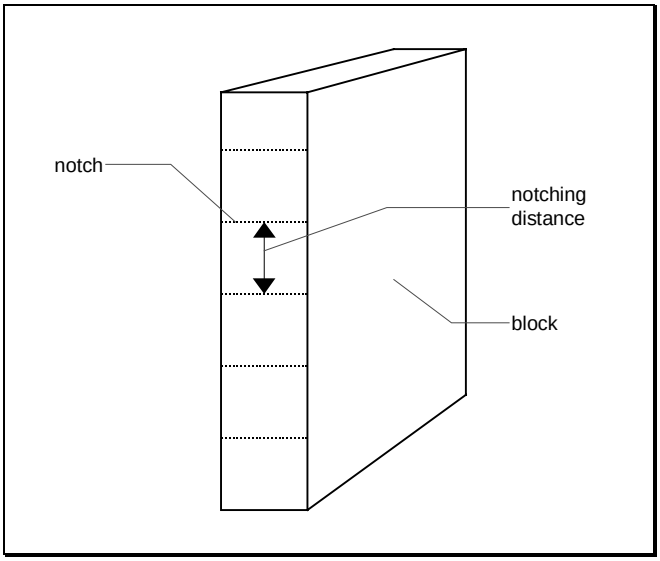



**Illustration 3-15: Parameters for Back Preparation Process**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/ProcessType** | name | required | This item identifies the back preparation process and must therefore always be set to **/BackPreparation**. |
| **/MillingDepth** | number | required | Milling depth |
| **/NotchingDistance** | number | optional | Notching distance. |
| **/NotchingDepth** | number | optional | Notching depth. |
| **/StartPosition** | number | required | Starting position of milling tool (along the Y-axis of the operation coordinate system) |
| **/WorkingLength** | number | required | Working length of milling operation. |

**Table 3-25: Dictionary for Back Preparation Process**

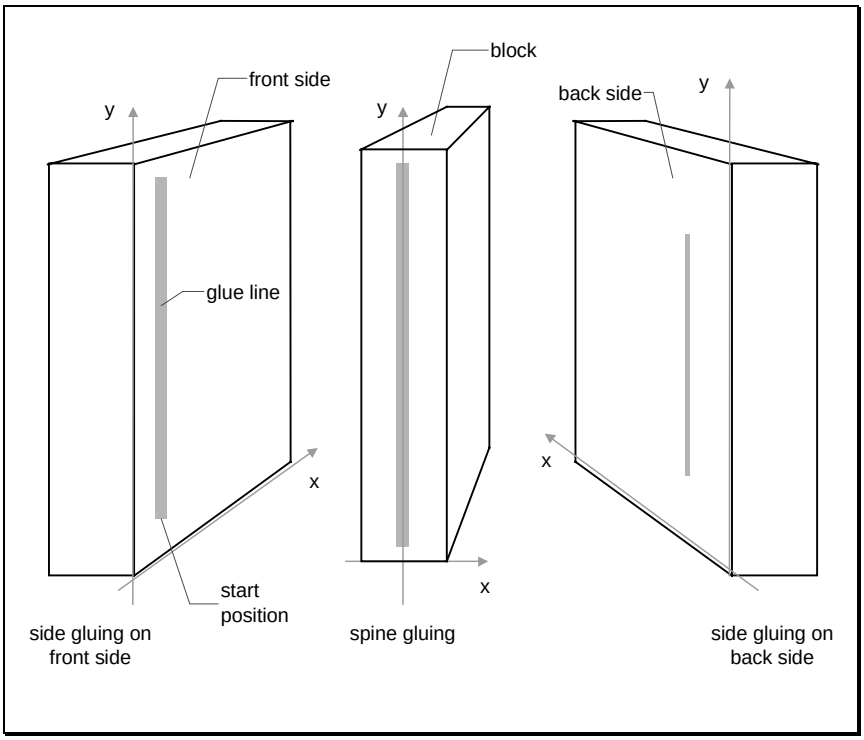



**Illustration 3-16: Parameters for Gluing Application**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/ProcessType** | name | required | This item identifies the glue application process and must therefore always be set to **/GlueApplication**. |
| **/GluingTechnique** | name | required | Type or technique of gluing application: **/SpineGluing**, **/SideGluingFront**, **/SideGluingBack** |
| **/GlueLine** | dictionary | required | GlueLine dictionary (see Table 3-32: “GlueLine Dictionary describing a Glue Line”). |

**Table 3-26: Dictionary for Glue Application**

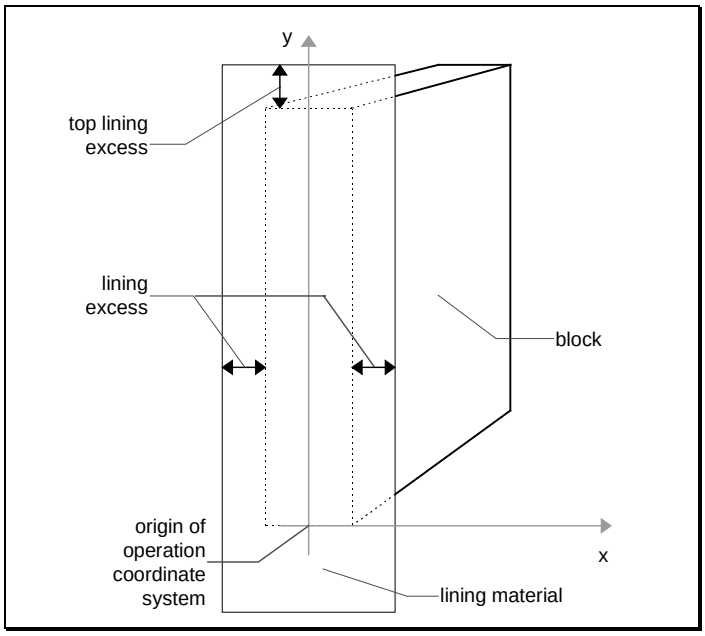



**Illustration 3-17: Parameters for Lining Process**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/ProcessType** | name | required | This item identifies the lining process and must therefore always be set to **/Lining**. |
| **/TopLiningExcess** | number | required | Top lining excess.<br>This value may be negative. |
| **/LiningExcess** | number | required | Lining excess on each (left and right) side. The lining material is assumed to be centered between left and right. |
| **/LiningLength** | number | required | Length of lining material along binding edge. |
| **/LiningMaterial** | name | optional | Lining material: **/Gauze**, **/Calico**, **/PaperlinedMules**, **/CrepePaper** |
| **/LiningBrand** | string | optional | Lining brand |

**Table 3-27: Dictionary for Lining Process**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/ProcessType** | name | required | This item identifies the cover application process and must therefore always be set to **/CoverApplication**. |
| **/CoverOffset** | array of 2 numbers | required | Position of cover in relation to book block given in the cover sheet coordinate system. |
| **/ScoringOffsets** | array of numbers | *see meaning* | Position of scoring given in the operation coordinate system.<br>If the *ScoringOffsets* attribute is present, the *ScoringSide* attribute must also be defined and vice versa. Both array must have the same number of elements. |
| **/ScoringSide** | array of names | *see meaning* | Specifies the side from which the scoring tool works: **/FromInside** or **/FromOutside**.<br>If the *ScoringSide* attribute is present, the *ScoringOffsets* attribute must also be defined and vice versa. Both array must have the same number of elements. |

**Table 3-28: Dictionary for Cover Applications**

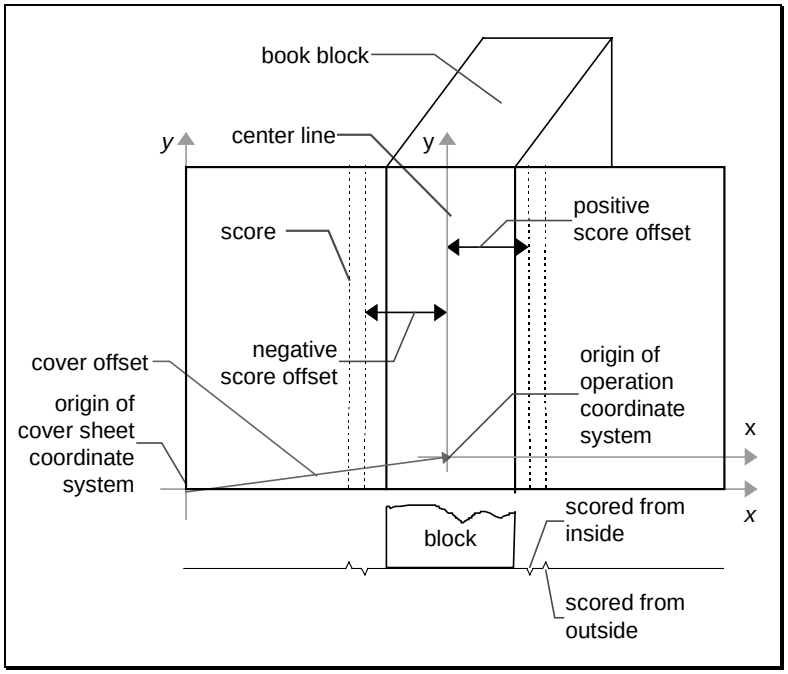



**Illustration 3-18: Parameters of Cover Application**

**Example 3-21: Adhesive Binding**

The example shown below specifies the following sequence of processes:

- back preparation
- spine gluing
- side gluing on front side
- side gluing on back side
- lining
- spine gluing
- cover application

```
CIP3BeginProductDefinition
/CIP3Products [
<<
   /CIP3ProductName (adhesive bound book)
   /CIP3ProductOperation /AdhesiveBinding
   /CIP3ProductParams  <<
                          /Processes
                          [
                              %% ----- back preparation
                              << /ProcessType /BackPreparation
                                 /MillingDepth 1.5 mm
                                 /NotchingDistance 2.5 cm
                                 /NotchingDepth 0.3 mm
                                 /StartPosition 5 mm
                                 /WorkingLength 275 mm
                              >>
                              %% ----- spine gluing (first time)
                              << /ProcessType /GlueApplication
                                 /GluingTechnique /SpineGluing
                                 /GlueLine <<
                                              /StartPosition [ 0 mm  5 mm ]
                                              /WorkingPath [ 0 mm 29.2 cm ]
                                              /GlueType /ColdGlue
                                              /GlueLineWidth 10 mm
                                           >>
                              >>
                              %% ----- side gluing on front side
                              << /ProcessType /GlueApplication
                                 /GluingTechnique /SideGluingFront
                                 /GlueLine <<
                                              /StartPosition [ 7 mm 15 mm ]
                                              /WorkingPath [ 7 mm 27.5 cm ]
                                              /GlueType /ColdGlue
                                              /GlueLineWidth 10 mm
                                           >>
                              >>
```

```
                              %% ----- side gluing on back side
                              << /ProcessType /GlueApplication
                                 /GluingTechnique /SideGluingBack
                                 /GlueLine <<
                                             /StartPosition [ 7 mm 30 mm ]
                                             /WorkingPath [ 0 mm 210 mm ]
                                             /GlueType /ColdGlue
                                             /GlueLineWidth 3 mm
                                          >>
                              >>
                              % ----- lining
                              << /ProcessType /Lining
                                 /TopLiningExcess 2.5 cm
                                 /LiningExcess 1.5 cm
                                 /LiningLength 35 cm
                                 /LiningMaterial /Gauze
                              >>
                              %% ----- spine gluing (second time)
                              << /ProcessType /GlueApplication
                                 /GluingTechnique /SpineGluing
                                 /GlueLine <<
                                             /StartPosition [ 0 mm  2 mm ]
                                             /WorkingPath [ 0 mm 295 mm ]
                                             /GlueType /ColdGlue
                                             /GlueLineWidth 10 mm
                                          >>
                              >>
                              % ----- cover application
                              << /ProcessType /CoverApplication
                                 /CoverOffset [ 23 cm 1 cm ]
                                 /ScoringOffsets [-3.3 cm -3 cm 3 cm 3.3 cm]
                                 /ScoringSide  [ /FromOutside
                                                 /FromInside
                                                 /FromInside
                                                 /FromOutside ]
                              >>
                           ]
                        >>
   /CIP3ProductComponents
   [
      <<
             /SourceType /PartialProduct
             /SourceProduct (book block)
             /Params << /Orientation [1 0 0 1 0 0] >>
      >>
      <<
             /SourceType /Sheet
             /SourceProduct (cover sheet)
             /Params << /Orientation [1 0 0 1 0 0] >>
```

```
      >>
   ]
>>


<<
   /CIP3ProductName (book block)
   % ... the definition of the book block operation would go here ...
>>
] def

/CIP3FinalProducts [ (adhesive bound book) ] def
CIP3EndProductDefinition
```

### 3.3.2.9. Trimming

This operation describes the trimming process. The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/Trimming**.

The operation requires one component, the partial product to be trimmed.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Width** | number | optional | width of trimmed product |
| **/Height** | number | optional | height of trimmed product |
| **/TrimmingOffset** | number | optional | amount to be cut at bottom side |

**Table 3-29: Operation specific dictionary for Trimming operation**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component.<br>The binding edge is assumed to be at the Y-axis after applying this matrix to the book block. |

**Table 3-30: Component specific dictionary for Trimming operation**

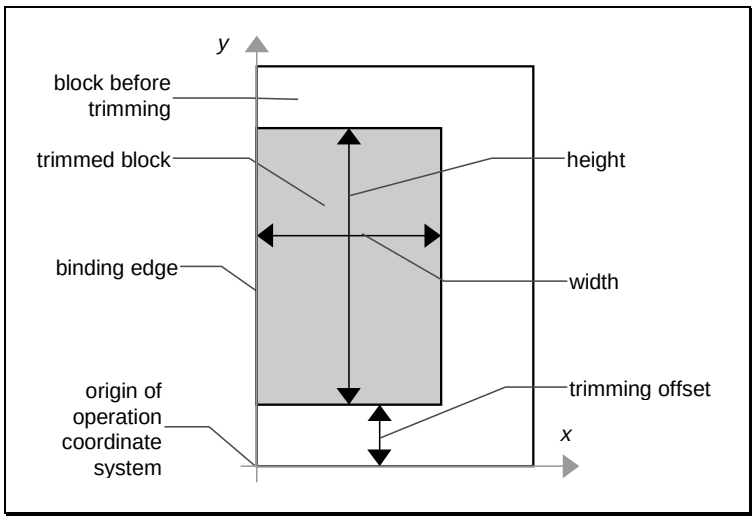



**Illustration 3-19: Trimming Parameters**

**Example 3-22: Trimming**

```
CIP3BeginProductDefinition
/CIP3Products [
<<
   /CIP3ProductName (trimmed block)
   /CIP3ProductOperation /Trimming
   /CIP3ProductParams <<
                         /Width  210 mm
                         /Height 297 mm
                         /TrimmingOffset  8 mm
                       >>
   /CIP3ProductComponents
   [
      <<
          /SourceType /PartialProduct
          /SourceProduct (book block)
          /Params << /Orientation [1 0 0 1 0 0] >>
      >>
   ]
>>

<<
   /CIP3ProductName (book block)
   % ... the definition of the book block would go here ...
>>
] def

/CIP3FinalProducts [ (trimmed block) ] def
CIP3EndProductDefinition
```

### 3.3.2.10. Gluing In

This operation describes the gluing-in process. The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/GluingIn**. The operation can be used to describe the gluing in of cards as well as the gluing in of samples, like a CD-ROM.

The operation requires two components in the following sequence:

- "mother" sheet
- sheet (e.g. card) to be glued in (optional).

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/SheetOffset** | array of 2 numbers | required | Offset between sheet to be glued in and "mother" sheet. |
| **/GlueLines** | array of dictionaries | required | Array of GlueLine dictionaries (see Table 3-32: “GlueLine Dictionary describing a Glue Line”). |
| **/Sample** | string | optional | Sample to be glued in. |

**Table 3-31: Operation specific dictionary for Gluing In operation**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/StartPosition** | array of 2 numbers | required | Start position of glue line . The start position is given in the coordinate system of the "mother" sheet. |
| **/WorkingPath** | array of 2 numbers | required | Relative working path of the gluing tool. |
| **/GlueType** | name | optional | Glue type: **/ColdGlue**, **/Hotmelt**, **/PUR** = Polyurethan |
| **/GlueBrand** | string | optional | Glue brand |
| **/GluingPattern** | array of 2 numbers *or* empty array | optional | Glue line pattern defined by the length of a glue line segment (1st array element) and glue line gap (2nd array element). An empty array specifies a solid glue line. |
| **/GlueLineWidth** | number | optional | Width of glue line. |

**Table 3-32: GlueLine Dictionary describing a Glue Line**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component.<br>The binding edge is assumed to be at the Y-axis after applying this matrix to the book block. |

**Table 3-33: Component specific dictionary for GluingIn operation**

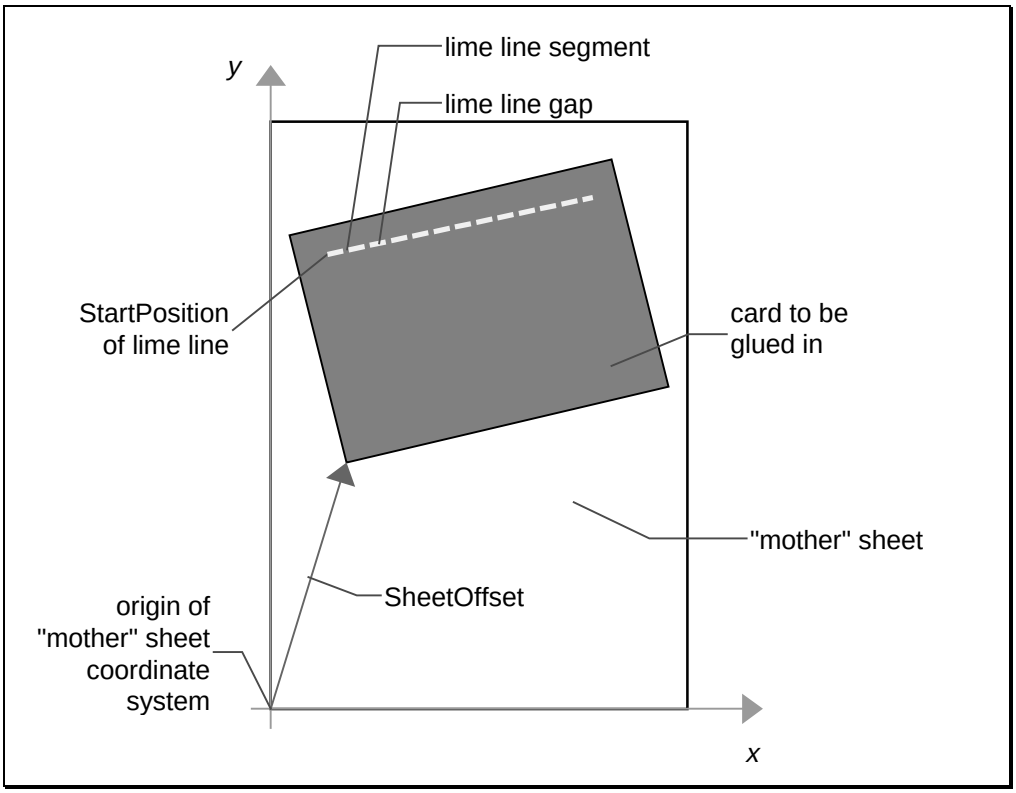



**Illustration 3-20: Parameters of Gluing In**

**Example 3-23: Gluing In**

```
CIP3BeginProductDefinition
/CIP3Products [
<<
   /CIP3ProductName (sheet with card)
   /CIP3ProductOperation /GluingIn
   /CIP3ProductParams <<
                          /SheetOffset [2 cm 15 cm ]
                          /GlueLines [
                                       << /StartPosition [ 3 cm 23 cm ]
                                          /WorkingPath [ 14.92 cm 28.18 cm ]
                                          % Angle 23.5, Length 13 cm
                                          /GlueType /Hotmelt
                                          /GluingPattern [ 2 cm 0.5 cm ]
                                       >>
                                       % second glue line would go here
                                     ]
                       >>
   /CIP3ProductComponents
   [
      <<
            % mother sheet
            /SourceType /Sheet
            /SourceSheet (mother sheet)
            /Params << /Orientation [1 0 0 1 0 0] >>
      >>
      <<
            % card to be glued in  (rotated 23.5 degrees to the left)
            /SourceType /Sheet
            /SourceSheet (card)
            /Params << /Orientation [0.91706 0.39875 –0.39875 0.91796 0.0 0.0] >>
      >>
   ]
>>
] def

/CIP3FinalProducts [ (sheet with card) ] def
CIP3EndProductDefinition
```

### 3.3.2.11. Folding

This operation describes the folding process. The way in which the folding operation is specified here is very similar to the way it is described in the *FoldProcedures* section (see chapter 3.10). The main difference is, that the folding operation described in the *FoldProcedures* section is restricted to the folding of a single sheet. In contrast, the folding operation specified here allows to fold a pile of sheets (or partial products) that have been gathered in a previous operation.

The operation type, which is specified by the *CIP3ProductOperation* attribute, must be set to **/Folding**.

The operation requires one component, the gathered sheets or partial products.

The following table shows the operations specific dictionary of the *Folding* operation. It is the same dictionary that is used for the folding procedures in chapter 3.10 (see Table 3-52: “Data dictionary for folding procedure” on page 95).

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3FoldDescription** | string | optional | Description of the type of fold according to Bundesverband Druck (Federal Association of Printing) i.e. general description.<br>*Hint: Within a PostScript string a parenthesis „( )“ must be marked with a ´\´ in front of it* |
| **CIP3FoldSheetIn** | array of 2 numbers | required | Input sheet format |
| **CIP3FoldProc** | array | required | Steps of folding procedure |

**Table 3-34: Operation specific dictionary for Folding operation**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Orientation** | matrix | required | Matrix describing the orientation of the component. |

**Table 3-35: Component specific dictionary for Folding operation**

**Example 3-24: Folding**

```
CIP3BeginProductDefinition
/CIP3Products [
<<
   /CIP3ProductName (folded sheets)
   /CIP3ProductOperation /Folding
   /CIP3ProductParams <<
                          /CIP3FoldDescription (F16 / 3W \(1/4\) + 1 ML)
                          /CIP3FoldSheetIn [880 mm 610 mm]
                          /CIP3FoldProc
                          [
                            220 mm 305 mm 660 mm 0 /Top Lime
                            220 mm /Front /Up Fold          % origin moves right
                            220 mm /Front /Up Fold          % origin moves right
                            220 mm /Front /Up Fold          % origin moves right
                              0 mm   8 mm 220 mm 0 /Top Cut % origin moves down
                              0 mm 594 mm 220 mm 0 /Top Cut
                            297 mm /Left /Up Fold           % origin moves up
                              5 mm 0 mm 0 297 mm /Top Cut   % origin moves right
                            210 mm 0 mm 0 297 mm /Top Cut
                          ]
                       >>
   /CIP3ProductComponents
   [
      <<
            /SourceType /PartialProduct
            /SourceSheet (gathered sheets)
            /Params << /Orientation [1 0 0 1 0 0] >>
      >>
   ]
>>

<<
   /CIP3ProductName (gathered sheets)
   % ... the definition of the gathered sheets would go here ...
>>
] def

/CIP3FinalProducts [ (folded sheets) ] def
CIP3EndProductDefinition
```

## 3.4. Administration Data

All administration data is stored in attributes, which should be defined in the *Sheet* structure or in the *Front* and *Back* structure.

| Attribute Name | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3AdmJobName** | string | required | Job name |
| **CIP3AdmJobCode** | string | optional | Code used for job identification |
| **CIP3AdmMake** | string | optional | Exposure device make |
| **CIP3AdmModel** | string | optional | Exposure device model |
| **CIP3AdmSoftware** | string | optional | Sheet creating program |
| **CIP3AdmCreationTime** | string | optional | Date and time of sheet creation. It is strongly recommended to use either the date format of the ctime() function, which is supported by many C run-time libraries, or the date format defined in PDF.<br>**ctime() Date Format**<br>This format of the ctime() function is supported by many C run-time libraries:<br>(*www MMM DD hh:mm:ss YYYY*)<br>**PDF Date Format**<br>(D:*YYYYMMDDhhmmssOHH'mm'*)<br>See page 133 of the PDF Reference Manual, Version 1.2 [PDF] for more detail. In contrast to the PDF Specification, the "D:" at the beginning of the date string is not optional in CIP3 PPF.<br>In both cases, the string should not contain any carriage-return or line-feed characters.<br>**Examples:**<br>(Thu Jan 28 12:25:12 1998)<br>(D:19980128122525+01'00') |
| **CIP3AdmArtist** | string | optional | Creator of the sheet |
| **CIP3AdmCopyright** | string | optional | Copyright |
| **CIP3AdmCustomer** | string | optional | Name of the target customer |
| **CIP3AdmSheetName** | string | *see meaning* | Name of the sheet. This attribute is required, if the CIP3 PPF file contains a PPFDirectory. The name must be a unique sheet name (at least within the whole CIP3 PPF file). It must not contain a slash character (or the Unicode equivalent character, if coded in Unicode). The string must be coded in up to 216 bytes (including the parentheses needed to build a PostScript string, even if it is coded in Unicode; see chapter 3.2). |
| **CIP3AdmSheetLay** | name | optional | Guiding edge of the paper in the press seen in the direction of paper flow: **/Left**, **/Right** |
| **CIP3AdmPrintVolume** | integer | optional | Print volume |

| Attribute Name | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3AdmFilmType** | string | optional | Brand and type of film material |
| **CIP3AdmPlateType** | string | optional | Brand and type of plate material |
| **CIP3AdmPaperGrade** | string | optional | Type of paper |
| **CIP3AdmPaperGrammage** | number | optional | Paper grammage in $g/m^2$ |
| **CIP3AdmPaperThickness** | number | optional | Paper thickness |
| **CIP3AdmPaperColor** | array of 3 numbers | optional | Paper color in CIE-L*a*b* |
| **CIP3AdmSeparationNames** | array of strings | required | Sequence of colors to be printed. The name of a separation must not contain a slash character (or the Unicode equivalent character, if coded in Unicode).<br>If the preview image is stored in separations, the sequence of the separation names must match the sequence of the *Separation* structures (see chapter 3.5.2).<br>If the preview image is stored as a composite preview image, the attribute must have the following value: `[(Cyan) (Magenta) (Yellow) (Black)]` (see chapter 3.5.1).<br>It is important that for standard process colors always the strings `(Cyan)`, `(Magenta)`, `(Yellow)`, and `(Black)` must be used. |
| **CIP3AdmInkInfo** | array of strings | optional | Information about used inks (e.g. exact name or manufacturer). The sequence of the ink information strings must match the sequence of the separations as specified by the *CIP3AdmSeparationNames* attribute. |
| **CIP3AdmInkColors** | array of array of 3 numbers | optional | Ink color specification in CIE-L*a*b*. The sequence of the ink color specification arrays must match the sequence of the separations as specified by the *CIP3AdmSeparationNames* attribute. |
| **CIP3AdmTypeOfScreen** | string | optional | (amplitude modulated) or (frequency modulated) |
| **CIP3AdmPSExtent** | array of 2 numbers | required | Extent of PostScript coordinate system. The PS extent specifies the exact size of the preview images (see chapter 3.5).<br>The extent can be specified in any of the units listed in Table 3-2: “Units in the CIP3 format”. |
| **CIP3AdmFilmTrf** | array of 6 numbers | optional | Transformation matrix used to map coordinates from the PostScript coordinate system to the film coordinate system.<br>If the transformation matrix is not defined, the default identity matrix `[1 0 0 1 0 0]` is used (see chapter 3.1.3). |
| **CIP3AdmFilmExtent** | array of 2 numbers | optional | Extent of film coordinate system.<br>The extent can be specified in any of the units listed in Table 3-2: “Units in the CIP3 format”.<br>If the extent is not defined, the value will be inherited from the *CIP3AdmPSExtent* attribute (see chapter 3.1.3). |

| Attribute Name | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3AdmPlateTrf** | array of 6 numbers | optional | Transformation matrix used to map coordinates from the film coordinate system to the plate coordinate system.<br>If the transformation matrix is not defined, the default identity matrix `[1 0 0 1 0 0]` is used (see chapter 3.1.3). |
| **CIP3AdmPlateExtent** | array of 2 numbers | optional | Extent of plate coordinate system.<br>The extent can be specified in any of the units listed in Table 3-2: “Units in the CIP3 format”.<br>If the extent is not defined, the value will be inherited from the *CIP3AdmFilmExtent* attribute (see chapter 3.1.3). |
| **CIP3AdmPressTrf** | array of 6 numbers | optional | Transformation matrix used to map coordinates from the plate coordinate system to the press coordinate system.<br>If the transformation matrix is not defined, the default identity matrix `[1 0 0 1 0 0]` is used (see chapter 3.1.3). |
| **CIP3AdmPressExtent** | array of 2 numbers | optional | Extent of press coordinate system.<br>The extent can be specified in any of the units listed in Table 3-2: “Units in the CIP3 format”.<br>If the extent is not defined, the value will be inherited from the *CIP3AdmPlateExtent* attribute (see chapter 3.1.3). |
| **CIP3AdmPaperTrf** | array of 6 numbers | optional | Transformation matrix used to map coordinates from the press coordinate system to the paper coordinate system.<br>If the transformation matrix is not defined, the default identity matrix `[1 0 0 1 0 0]` is used (see chapter 3.1.3). |
| **CIP3AdmPaperExtent** | array of 2 numbers | optional | Extent of paper coordinate system.<br>The extent can be specified in any of the units listed in Table 3-2: “Units in the CIP3 format”.<br>If the extent is not defined, the value will be inherited from the *CIP3AdmPressExtent* attribute (see chapter 3.1.3). |

**Table 3-36: General administration data**

In addition to the general administration data there are some attributes that can be used to specify information used for web presses.

| Attribute Name | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3AdmPaperSource** | name | optional | Input source of paper: **/Reel** or **/Sheet** |
| **CIP3AdmPaperDestination** | name | optional | Output destination of paper: **/Reel**, **/Sheet** or **/Folded** |
| **CIP3AdmTintingColorName** | string | optional | Name of tinting color |
| **CIP3AdmTintingColor** | array of 3 numbers | optional | Tinting color specification in CIE-L*a*b* |
| **CIP3AdmReelWidth** | number | optional | Reel width (for rotary printing) |
| **CIP3AdmCylinderCircumference** | number | optional | Cylinder circumference (for rotary printing) |
| **CIP3AdmCarbonizingColorName** | string | optional | Name of color used for carbonizing |
| **CIP3AdmCoating** | string | optional | Type of coating |
| **CIP3AdmLongitudinalApp** | array of dictionaries | optional | Specification of longitudinal slitting, perforating, gluing, or stitching (see Table 3-38: “Data dictionary for longitudinal and cross applications”). This item is only used for reel processing. |
| **CIP3AdmCrossApp** | array of dictionaries | optional | Specification of cross cutting, perforating, or gluing (see Table 3-38: “Data dictionary for longitudinal and cross applications”). This item is only used for reel processing. |

**Table 3-37: Web press data**

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/ReferenceEdge** | name | required | Reference edge: **/Left** or **/Right** (seen in direction of web travel) |
| **/Travel** | number | required | Position of tool in direction perpendicular to working path. For a longitudinal application the travel is the distance of the tool from the reference edge. For a cross application the travel is the distance of the tool from the front edge. |
| **/StartPosition** | number | required | Starting position of tool in direction of working path. For a longitudinal application the travel is the distance of the tool from the front edge. For a cross application the travel is the distance of the tool from the reference edge. |
| **/WorkingLength** | number | required | Length of working path. For a longitudinal application the working path is always parallel to the direction of web travel. For a cross application the working path is always parallel to the front edge) |
| **/Function** | name | required | Type of function: **/Cut** = slitting, **/Perforate** = perforating, **/Glue** = gluing, **/Stitch** = stitching |

**Table 3-38: Data dictionary for longitudinal and cross applications**

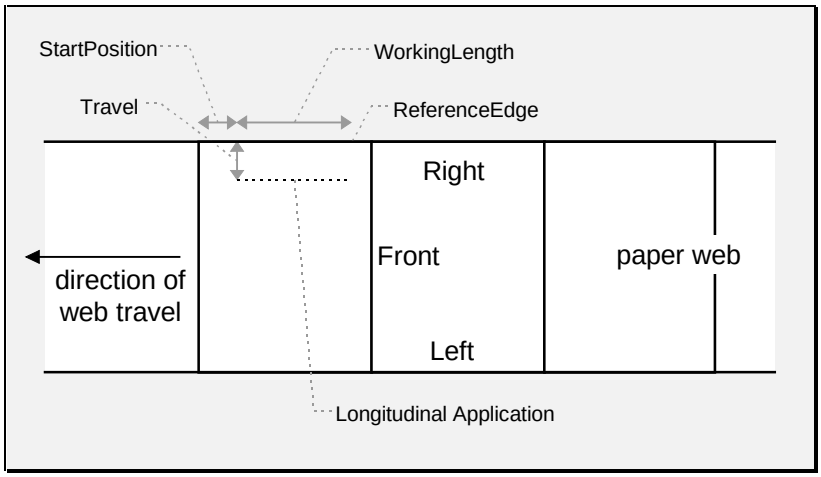



**Illustration 3-21: Example of longitudinal application (view from top)**

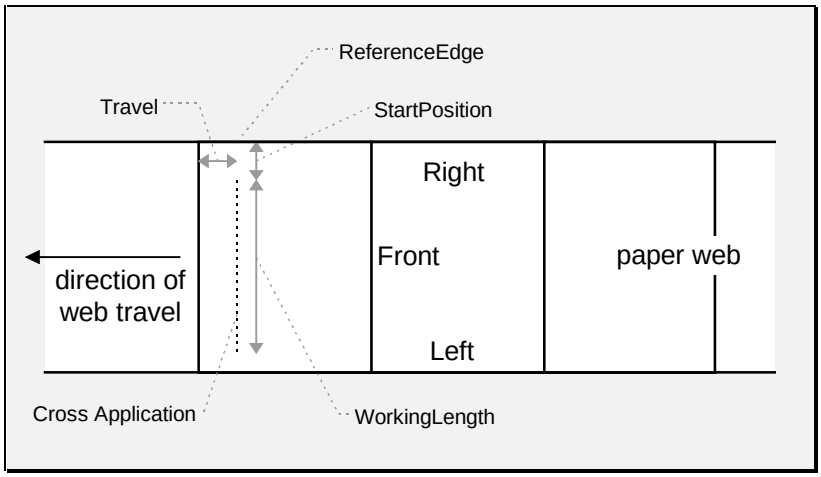



**Illustration 3-22: Example of cross application (view from top)**

**Example 3-25: Some Administration Data Attributes**

```
/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
/CIP3AdmPSExtent [100 cm 70 cm] def
/CIP3AdmSheetLay Left def
/CIP3AdmPrintVolume 120000 def
/CIP3AdmPaperGrammage 130 def
/CIP3AdmSeparationNames [ (Black) (Cyan) (Magenta) (Yellow) ] def
```

## 3.5. Continuous Tone Image in Reduced Resolution

The ***PreviewImage*** structure includes the contents of the complete sheet as low resolution continuous tone image. If only the standard printing colors Cyan, Magenta, Yellow, and Black are used, it is possible to store the image as a composite CMYK image. If more or other colors are used, it is required to store separate images for each color separation. In this case it is also allowed to store a bitonal image of a higher resolution instead of a continuous tone image.

In order to reduce the size of the CIP3 PPF file, some of the encoding and compression techniques specified in PostScript can be applied to the image data. For a complete description please refer to the PostScript Language Reference Manual (also known as Red Book [PostScript], chapter 3.13.3: Details of Individual Filters). The **/Binary** encoding type, which is not specified in the Red Book, is used for raw binary data without any special encoding applied.

It is neither required nor possible to specify the image read procedure. This procedure will be built implicitly during the interpretation of the CIP3 PPF file.

In order to create image data for the CIP3 PPF file, first the compression technique must be applied. The resulting data must then be encoded as specified. When reading a CIP3 PPF, the two mechanisms have to be applied in the opposite order.

The extent of the PostScript coordinate system as specified by the *CIP3AdmPSExtent* attribute, the resolution of the preview image, and width and height of the image must fulfill the following requirements:

```
x-ps-extent / 72 * x-resolution = width ± 1
y-ps-extent / 72 * y-resolution = height ± 1
```

| Attribute Name | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3PreviewImageWidth** | integer | required | Width of preview image in pixel |
| **CIP3PreviewImageHeight** | integer | required | Height of preview image in pixel |
| **CIP3PreviewImageBitsPerComp** | integer | required | Number of bits per color component (1 or 8) |
| **CIP3PreviewImageComponents** | integer | required | Number of image components (1 or 4) |
| **CIP3PreviewImageMatrix** | array of 6 numbers | required | Image matrix (see chapter 3.5.3) |
| **CIP3PreviewImageResolution** | array of 2 numbers | required | Resolution of preview image in pixels per inch |
| **CIP3PreviewImageEncoding** | name | required | Type of encoding: **/Binary**, **/ASCIIHexDecode**, **/ASCII85Decode** |
| **CIP3PreviewImageCompression** | name | required | Type of compression: **/None**, **/RunLengthDecode**, **/DCTDecode** (= JPEG), **/CCITTFaxDecode** (= Fax Group 3 and 4) |
| **CIP3PreviewImageFilterDict** | dictionary | *see meaning* | Dictionary containing special information for compression method (see Red Book [PostScript], page 127).<br>This dictionary entry is required, if either **/DCTDecode** or **/CCITTFaxDecode** compression is used. The dictionary may be empty. |
| **CIP3PreviewImageByteAlign** | integer | optional | Specifies that the image data for one image line is aligned to the number of bytes given (1, 2, or 4; default is 1)<br>This attribute may only be used, if no encoding or compression is used. |
| **CIP3PreviewImageDataSize** | integer | optional | Specifies the number of bytes used to store the compressed and encoded image (as specified in the attributes *CIP3PreviewImageCompression* and *CIP3PreviewImageEncoding*). See chapter 3.5.5 for detailed description of where the image data starts. |

**Table 3-39: Attributes of the preview image**

### 3.5.1. Composite Preview Image

If the standard printing colors Cyan, Magenta, Yellow and Black (in this order) are used, it is possible to store the image as a composite CMYK image. In this case the *CIP3PreviewImageComponents* attribute has to be set to 4.

A color component value of 0 represents no ink, while a value of 255 represents full ink (see DeviceCMYK color model in chapter 4.8.2. of the PostScript Language Reference Manual [PostScript]).

**Example 3-26: Composite Preview Image (CMYK)**

```
CIP3BeginPreviewImage
/CIP3PreviewImageWidth 2000 def
/CIP3PreviewImageHeight 1400 def
/CIP3PreviewImageBitsPerComp 8 def
/CIP3PreviewImageComponents 4 def
/CIP3PreviewImageMatrix [2000 0 0 -1400 0 1400] def
/CIP3PreviewImageResolution [ 50.8 50.8 ] def
/CIP3PreviewImageEncoding /ASCIIHexDecode def
/CIP3PreviewImageCompression /DCTDecode def
/CIP3PreviewImageFilterDict <<>> def
CIP3PreviewImage
... <image data>
CIP3EndPreviewImage
```

### 3.5.2. Preview Image with Separations

If the preview image is stored in separations, a substructure for each separation must be specified in the CIP3 file. The sequence of the separations must match the sequence specified in the *CIP3AdmSeparationNames* attribute (see Table 3-36: “General administration data”). In addition, the *CIP3PreviewImageComponents* attribute must be set to 1.

When the image is coded with 1 bit per component, a gray value of 0 represents full ink, while a value of 1 represents no ink. When the image is coded with 8 bits per component, a gray value of 0 represents full ink, while a value of 255 represents no ink (see DeviceGray color model chapter 4.8.2. of the PostScript Language Reference Manual [PostScript]).

**Example 3-27: Preview Image with two Separations**

```
CIP3BeginPreviewImage
CIP3BeginSeparation
/CIP3PreviewImageWidth 2000 def
/CIP3PreviewImageHeight 1400 def
/CIP3PreviewImageBitsPerComp 8 def
/CIP3PreviewImageComponents 1 def
/CIP3PreviewImageMatrix [2000 0 0 1400 0 0] def
/CIP3PreviewImageResolution [ 50.8 50.8 ] def
/CIP3PreviewImageEncoding /ASCII85Decode def
/CIP3PreviewImageCompression /RunLengthDecode def
CIP3PreviewImage <... runlength compressed and ASCII85 encoded image data of
                     first separation ...>
CIP3EndSeparation
CIP3BeginSeparation
/CIP3PreviewImageWidth 2000 def
/CIP3PreviewImageHeight 1400 def
/CIP3PreviewImageBitsPerComp 8 def
/CIP3PreviewImageComponents 1 def
```

```
/CIP3PreviewImageMatrix [2000 0 0 1400 0 0] def
/CIP3PreviewImageResolution [ 50.8 50.8 ] def
/CIP3PreviewImageEncoding /Binary def
/CIP3PreviewImageCompression /None def
CIP3PreviewImage <... raw image data of second separation; should be separated
                  from CIP3PreviewImage keyword by one single space character ...>
CIP3EndSeparation
CIP3EndPreviewImage
```

### 3.5.3. Sequence of Image Data

In PostScript it is very common to represent the image data line by line, starting with the first pixel in the lower-left corner. Within the CIP3 PPF eight different orientations are allowed. Depending on the orientation used for the generation of the image data, one of the PostScript matrices shown in the table below has to be defined in the *CIP3PreviewImageMatrix* attribute. In fact "w" and "h" have to be substituted by the width and the height of the preview image.

| Sequence of Data | Fast Running Index | Slow Running Index | PostScript Matrix |
|---|---|---|---|
|  | from left to right | from bottom to top | [w 0 0 h 0 0] |
|  | from left to right | from top to bottom | [w 0 0 -h 0 h] |
|  | from right to left | from bottom to top | [-w 0 0 h w 0] |
|  | from right to left | from top to bottom | [-w 0 0 -h w h] |
|  | from bottom to top | from left to right | [0 h w 0 0 0] |
|  | from top to bottom | from left to right | [0 h -w 0 h 0] |
|  | from bottom to top | from right to left | [0 -h w 0 0 w] |
|  | from top to bottom | from right to left | [0 -h -w 0 h w] |

**Table 3-40: Orientation of image data**

The first orientation specified in the table shown above is the standard orientation for PostScript images. If one of the other image orientations is used, it is important that only the orientation in which the image is stored in the CIP3 PPF file is changed. The only attribute that reflects this change is the *CIP3PreviewImageMatrix* attribute. Width and height must be given in the "original" orientation and not in the orientation in which the image is stored in the file.

### 3.5.4. Rules for the Generation of the Preview Image

In order to be useful for the ink consumption calculation, the preview data must be generated with an appropriate resolution. This does not only mean spatial resolution, but also color or tonal resolution. Spatial resolution is important for thin lines, while tonal resolution becomes important with large areas filled with a certain tonal value.

The maximum error caused by limited spatial and tonal resolution should be less than 1 %.

#### Spatial Resolution

Since some pixel of the preview image might fall on the border between two zones, their tonal values have to be split up. In the worst case the pixels are falling just in the middle between a totally white and a totally black zone. In this case the tonal value will be 50 %, but only 25 % contribute to the black zone. Depending on the resolution of the preview image and the zone width the maximum error can be calculated as:

$$\mathrm{error}[\%] = \frac{100}{4 * \mathrm{resolution}[L / mm] * \mathrm{zone\_width}[mm]}$$

For zone width broader than 25 mm a resolution of 2 lines per mm will always result in an error less than 0.5 %. Therefore a resolution of 2 lines per mm (equals 50.8 dpi) is suggested.

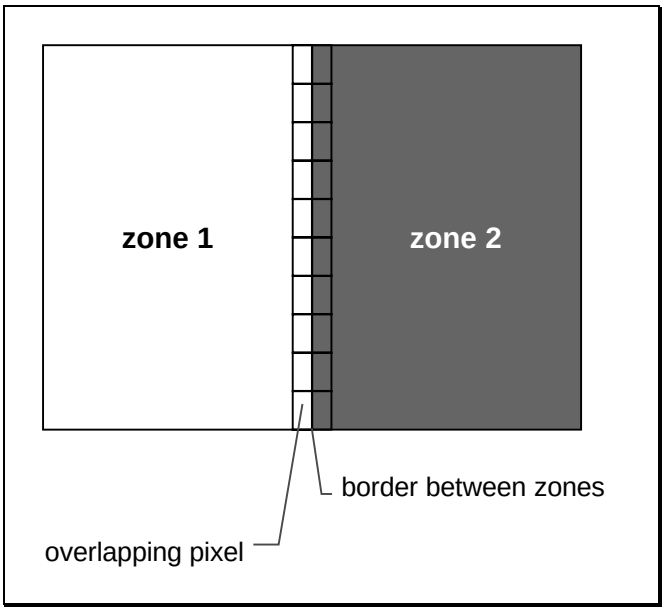



**Illustration 3-23: Worst case for area coverage calculation**

#### Tonal Resolution

The error caused by color quantization depends on the number of shades available. If the real tonal value is rounded to the closest (lower or higher) available shade, the error caused by this can be calculated by:

$$error[\%] = \frac{100}{2 * number\_of\_shades}$$

Therefore at least 64 shades should be used.

### Line Art Resolution

When rasterizing line art elements, the minimal line width is 1 pixel, i.e. 1/resolution. Therefore the relation between the printing resolution and the (spatial) resolution of the preview image is important for these kind of elements. In addition to that a specific characteristic of PostScript RIPs adds another error: within PostScript each pixel is set, which is touched by a line.

Tests with different PostScript jobs have shown, that a line art resolution of more than 300 dpi normally is sufficient for the ink consumption calculation.

### Conclusion

There are different ways to meet the requirements listed above. One way could be the following:

1. ripping the job with 406.4 dpi monochrome
2. filtering (with anti-aliasing) the image data down by a factor of 8 in both directions. This results in an image of 50.8 dpi with 65 color shades.

Another way might be to filter (with anti-aliasing) the high resolution data (e.g. used for plate making):

1. taking the ripped data (2540 dpi monochrome)
2. filtering (with anti-aliasing) the image data down by a factor of 50 in both directions. This results in an image of 50.8 dpi with 2501 color shades
3. mapping 2501 shades to 256 shades (without affecting the spatial resolution)

**Note:** *Rasterizing a job with 50.8 dpi and 256 shades of gray is not sufficient. The problem in this case is the rendering of thin lines (see: Line Art Resolution*).

### Minimal Requirements

It is strongly recommended to adhere to the requirements described above:

- resolution of ripped line art at least 300 dpi
- spatial resolution of preview image approximately 20 pixel/cm (= 50.8 dpi)
- tonal resolution of preview image at least 64 shades

### 3.5.5. Using Image Encoding and Image Compression

The details of individual filters as described in chapter 3.13.3 of the PostScript Language Reference Manual [PostScript] must be obeyed when using encoding or compression. In particular the end of data codes must be provided after the image data: ">" for **/ASCIIHexDecode**, "~>" for **/ASCII85Decode**, and $128_{dec}$ for **/RunLengthDecode**.

#### End-Of-Line Recommendation for Binary Encoding

When using the **/Binary** image encoding, it is required that there is exactly one white space character between the *CIP3PreviewImage* keyword and the image data. Within PostScript all three ordinary forms of end-of-line are treated as one single white space character (see chapter 3.8.1 of the PostScript Language Reference Manual [PostScript] for a more detailed description):

- line-feed only (UNIX)
- carriage-return only (Macintosh)
- carriage-return followed by line-feed (Windows and MS-DOS)

Special attention must be paid, if carriage-return only is used as EOL convention (as on Macintosh computers). Since the first character of the image data might be a line-feed character, it is strongly recommended to use a space character instead of the EOL-character to separate the *CIP3PreviewImage* keyword from the image data. Although, this recommendation is only relevant to image data using the **/Binary** encoding

## 3.6. Characteristic Curves for Transfer

The characteristic curves for transfer are stored as attributes. They are the basis of the calculation of ink consumption.

| Attribute Name | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3TransferFilmCurveData** | array of numbers | required | Data of copy-to-film curve of transfer: an even number of values between 0.0 and 1.0; each pair defines one point of the transfer curve. |
| **CIP3TransferPlateCurveData** | array of numbers | required | Data of copy-to-printing-plate curve of transfer: an even number of values between 0.0 and 1.0; each pair defines one point of the transfer curve. |

**Table 3-41: Characteristic curves for transfer**

If the same values of the *CIP3TransferFilmCurveData* and *CIP3TransferPlateCurveData* attributes are valid for both front and back, the definition of the two attributes should be made in the *Sheet* structure of the CIP3 file. If there are different curves of transfer used for front and back, the definitions must be placed in the *Front* and the *Back* structure. If each separation uses different curves of transfer, the definitions can also be placed in the *Separation* structures.

If used for a Computer-to-Plate (CTP) process, the copy-to-film transfer curve can be set to the identity transfer curve by specifying the following array: `[0.0 0.0 1.0 1.0]`

The two characteristic curves for transfer are used to calculate the real area coverage from the data of the CIP3 preview images. If both transfer curves are identity transfer curves, each separation of a preview image can directly be interpreted as an arrays of area coverage values. Otherwise, the two transfer curves have to be applied one after another (see Example 3-28: “Curves of Transfer”).

**Example 3-28: Curves of Transfer**

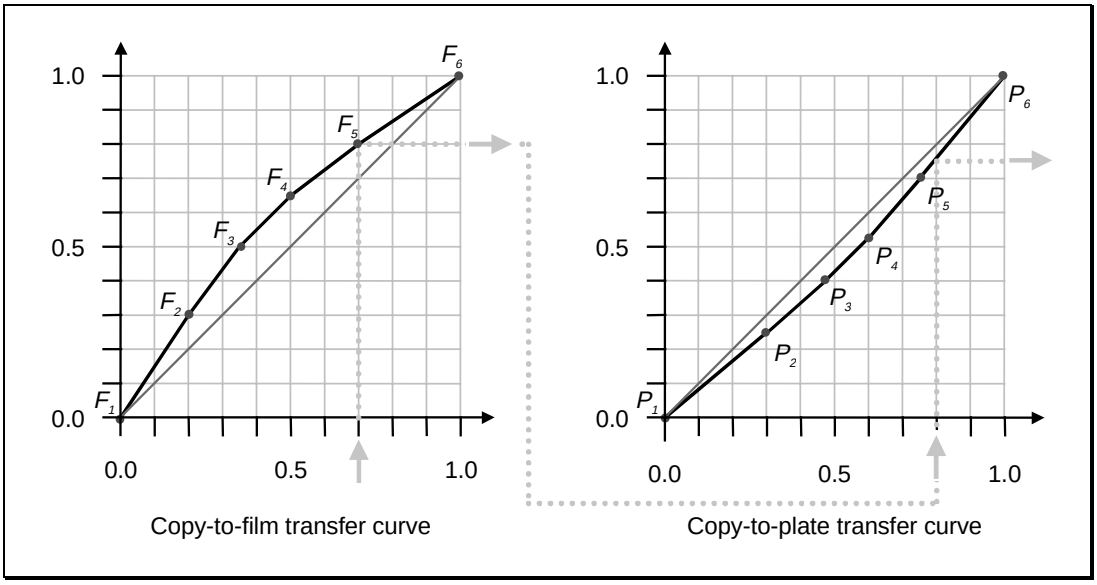



**Illustration 3-24: Curves of transfer example**

| Copy-to-Film | | |
|---|---|---|
| | In | Out |
| $F_1$ | 0.0 | 0.0 |
| $F_2$ | 0.2 | 0.3 |
| $F_3$ | 0.35 | 0.5 |
| $F_4$ | 0.5 | 0.65 |
| $F_5$ | 0.7 | 0.8 |
| $F_6$ | 1.0 | 1.0 |

| Copy-to-Plate | | |
|---|---|---|
| | In | Out |
| $P_1$ | 0.0 | 0.0 |
| $P_2$ | 0.3 | 0.25 |
| $P_3$ | 0.475 | 0.4 |
| $P_4$ | 0.6 | 0.45 |
| $P_5$ | 0.75 | 0.7 |
| $P_6$ | 1.0 | 1.0 |

**Table 3-42: Definition of example transfer curves**

**CIP3 PPF Sample code:**

```
/CIP3TransferFilmCurveData  [ 0.0   0.0     0.2   0.3     0.35   0.5
                              0.5   0.65    0.7   0.8     1.0    1.0  ] def

/CIP3TransferPlateCurveData [ 0.0   0.0     0.3   0.25    0.475  0.4
                              0.6   0.525   0.75  0.7     1.0    1.0  ] def
```

**Calculation example:**

A pixel value of 179 can be interpreted as 70% area coverage. Applying the copy-to-film transfer curve (at point $F_5$) results in an area coverage value of 80%. Applying the copy-to-plate transfer curve requires an interpolation between points $P_5$ and $P_6$ and finally results in an area coverage of 74.2%.

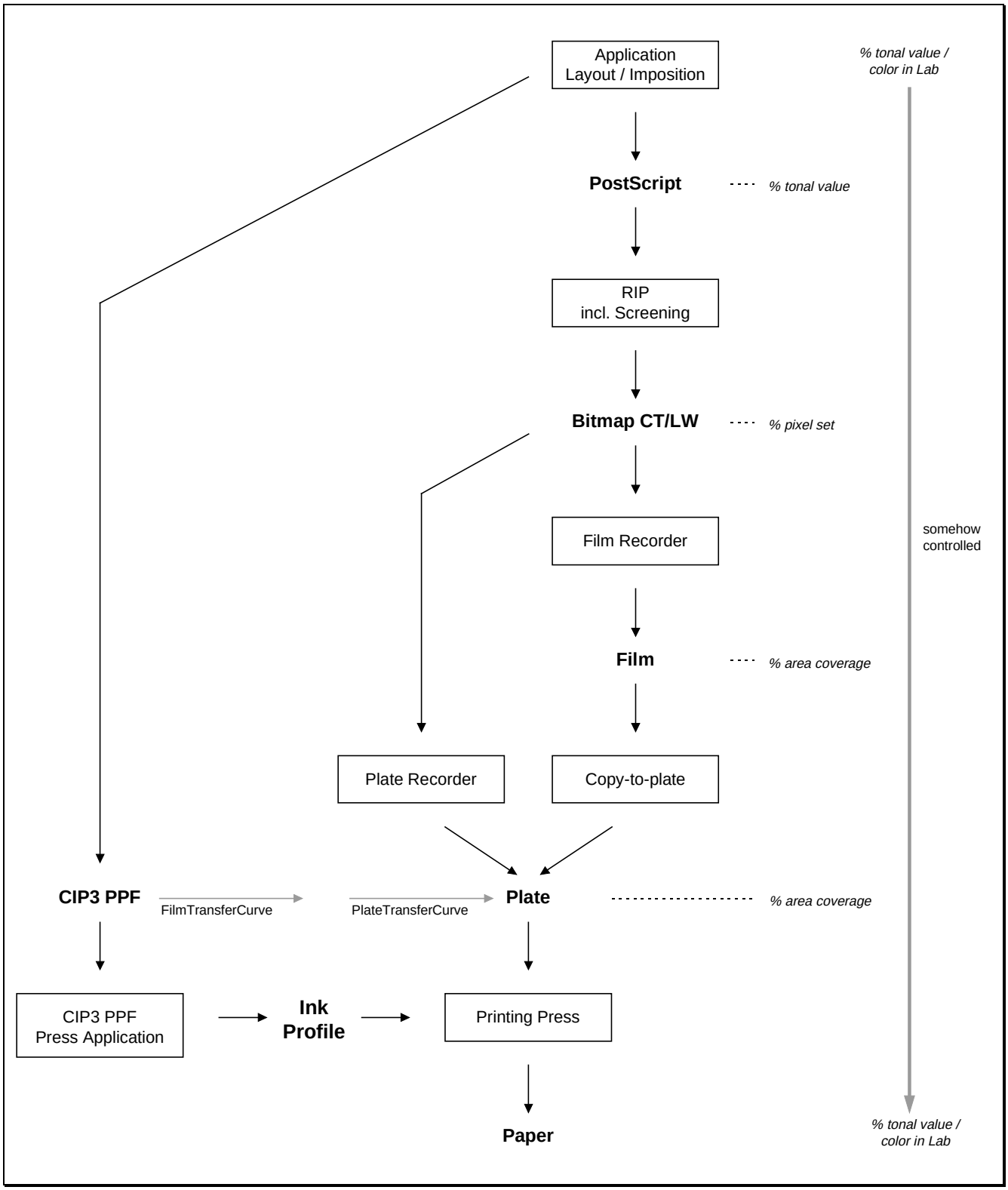



**Illustration 3-25: Curves of transfer**

## 3.7. Register Marks

All register marks are compiled in the ***RegisterMarks*** structure. The position and rotation of each register mark can be specified with the help of three parameters. It is important that the register marks are defined in such a way that their centers are on the point of origin of the coordinate system, because otherwise they are not positioned properly.

In addition to their position on the sheet register marks can be associated with a set of separations. By default each register mark is associated with each separation of the job. The ***CIP3SetRegisterMarkSeparations*** command is used to specify the set of separations to which the subsequently defined marks are bound.

| Parameter | Type | Meaning |
|---|---|---|
| ***separations*** | array of strings | List of separation names. Each name must exactly match one of the separations names specified in the *CIP3AdmSeparationNames* field (see Table 3-36: “General administration data”). |

**Table 3-43: Parameter for the CIP3SetRegisterMarkSeparations command**

| Parameter | Type | Meaning |
|---|---|---|
| ***translate-x*** | number | Relocation in X-direction. |
| ***translate-y*** | number | Relocation in Y-direction. |
| ***rotation*** | number | Rotation in degrees: positive graduation figures indicate counter-clockwise rotation; negative figures indicate clockwise rotation. |
| ***type*** | name | Type of register mark |

**Table 3-44: Parameter for the CIP3PlaceRegisterMark command**

**Example 3-29: Register Marks**

```
CIP3BeginRegisterMarks
30.0 30.0 0 /regm1 CIP3PlaceRegisterMark
625.0 30.0 0 /regm1 CIP3PlaceRegisterMark
625.0 872.0 0 /regm1 CIP3PlaceRegisterMark
30.0 872.0 0 /regm1 CIP3PlaceRegisterMark
[(Cyan) (Magenta) (Yellow)] CIP3SetRegisterMarkSeparations
5.0 30.0 0 /special_mark CIP3PlaceRegisterMark
CIP3EndRegisterMarks
```

## 3.8. Color and Ink Control

All color and ink measuring fields or control strips are compiled in the ***ColorControl*** structure. There are three different types of data elements defined:

- color measuring fields,
- density measuring fields, and
- color control strips.

### Color and Density Measuring Fields

Color and density measuring fields can be positioned on the front or back page of the sheet by using the ***CIP3PlaceMeasuringField*** command.

| Parameter | Type | Meaning |
|---|---|---|
| ***position-x*** | number | Position of the center of the color measuring field in X-direction. |
| ***position-y*** | number | Position of the center of the color measuring field in Y-direction. |
| ***data*** | dictionary | Dictionary containing detailed information about color measuring field (see Table 3-46: “Data dictionary entries for a color measuring field” and Table 3-47: “Data dictionary entries for a density measuring field”). |

**Table 3-45: Parameter for the CIP3PlaceMeasuringField command**

To allow for an easy extension of the color and ink control features of the CIP3 format the specific data required by a certain type of measuring field is collected in a PostScript dictionary. Each entry in a dictionary is build by a pair of key and value. Within CIP3 always names are used as keys.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/CIE-L*** | number | required | Color specification: value of L* |
| **/CIE-a*** | number | required | Color specification: value of a* |
| **/CIE-b*** | number | required | Color specification: value of b* |
| **/Diameter** | number | required | Diameter of measuring field |
| **/Light** | name | required | Type of light: **/D50**, **/D65**, ... |
| **/Observer** | int | required | Observer in degree (2 or 10) |
| **/Tolerance** | number | required | Tolerance in ΔE |
| **/Setup** | string | optional | Description of measurement setup |
| **/Percentages** | array of numbers | optional | Film percentage values for each separation. The number of array elements must match the number of separations (as defined by the *CIP3AdmSeparationNames* attribute) |
| **/ScreenRuling** | array of numbers | optional | Screen ruling values in lines per inch for each separation. The number of array elements must match the number of separations (as defined by the *CIP3AdmSeparationNames* attribute) |
| **/ScreenShape** | string | optional | shape of screening dots |
| **/DensityStandard** | string | optional | Density filter norm: **/DIN16536**, **/DIN16536NB**, **/ANSIA**, /**ANSIT** |
| **/Type** | name | required | Type of measuring: **/CIELAB** |

**Table 3-46: Data dictionary entries for a color measuring field**

**Example 3-30: Color measuring field**

```
100.0 mm 15.5 mm
  << /CIE-L*   62.0
     /CIE-a*  -31.0
     /CIE-b*  -48.0
     /Diameter 4.7.0 mm
     /Light /D65
     /Observer 2
     /Tolerance 5.0
     /DensityStandard /DIN16536
     /Type /CIELAB
  >> CIP3PlaceMeasuringField
```

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **/Screen** | string | required | Description of screen |
| **/Separation** | string | required | Reference to separation. This string must match one of the separation names given in the *CIP3AdmSeparationNames* attribute. |
| **/DensityBlack** | number | required | Density value measured with filter for black |
| **/ToleranceBlack** | array of 2 numbers | required | Upper and lower tolerance (in density units) |
| **/DensityCyan** | number | required | Density value measured with filter for Cyan |
| **/ToleranceCyan** | array of 2 numbers | required | Upper and lower tolerance (in density units) |
| **/DensityMagenta** | number | required | Density value measured with filter for Magenta |
| **/ToleranceMagenta** | array of 2 numbers | required | Upper and lower tolerance (in density units) |
| **/DensityYellow** | number | required | Density value measured with filter for Yellow |
| **/ToleranceYellow** | array of 2 numbers | required | Upper and lower tolerance (in density units) |
| **/DotGain** | number | required | Percentage of dot gain |
| **/ToleranceDotGain** | array of 2 numbers | required | Upper and lower tolerance (in percentage) |
| **/Percentage** | number | required | Film percentage or equivalent |
| **/Diameter** | number | required | Diameter of measuring field |
| **/Setup** | string | optional | Description of measurement setup |
| **/Type** | name | required | Type of measuring field: **/Density** |

**Table 3-47: Data dictionary entries for a density measuring field**

**Example 3-31: Density measuring field**

```
100.0 mm 15.5 mm
  << /Screen (60 / cm)
     /Separation (Black)
     /DensityBlack 0.331      /ToleranceBlack [-0.02 0.02]
     /DensityCyan 0.293       /ToleranceCyan [-0.02 0.02]
     /DensityMagenta 0.313    /ToleranceMagenta [-0.02 0.02]
     /DensityYellow 0.054     /ToleranceYellow [-0.02 0.02]
     /DotGain 15.2            /ToleranceDotGain [2.0 -2.0]
     /Percentage 50.0
     /Diameter 5.0 mm
     /Type /Density
  >> CIP3PlaceMeasuringField
```

### Color Control Strips

A color control strip can be specified with the ***CIP3PlaceColorControlStrip***. Positioning and Rotation of control strips refer to the point of origin of the coordinate system in which the strip is defined.

| Parameter | Type | Meaning |
|---|---|---|
| ***position-x*** | number | Position of the center of the control strip in X-direction. |
| ***position-y*** | number | Position of the center of the control strip in Y-direction. |
| ***rotation*** | number | Rotation in degrees: positive graduation figures indicate counter-clockwise rotation; negative figures indicate clockwise rotation. |
| ***width*** | number | Width of control strip (before rotation). |
| ***height*** | number | Height of control strip (before rotation). |
| ***data*** | array of arrays | Definition of fields contained in the control strip. This array may be empty, if no definition of the control strip is needed. The lower left corner of the control strip box (defined by position, width, and height) builds the origin of the coordinate system used for the definition of the elements contained in the control strip. |
| ***name*** | name | Name of control strip |

**Table 3-48: Parameter for the CIP3PlaceColorControlStrip command**

**Example 3-32: Color and Ink Control**

```
CIP3BeginColorControl
/Black50 <<
   /Screen (60 / cm)
   /Separation (Black)
   /DensityBlack 0.331
   /ToleranceBlack [-0.02 0.02]
   /DensityCyan 0.293
   /ToleranceCyan [-0.02 0.02]
   /DensityMagenta 0.313
   /ToleranceMagenta [-0.02 0.02]
   /DensityYellow 0.054
   /ToleranceYellow [-0.02 0.02]
   /DotGain 15.2
   /ToleranceDotGain [2.0 -2.0]
   /Percentage 50.0
   /Diameter 5.0 mm
   /Type /Density
>> def
/Cyan50 <<
   /Screen (60 / cm)
   /Separation (Cyan)
   /DensityBlack 0.338
   /ToleranceBlack [-0.02 0.02]
```

```
   /DensityCyan 0.446
   /ToleranceCyan [-0.02 0.02]
   /DensityMagenta 0.043
   /ToleranceMagenta [-0.02 0.02]
   /DensityYellow 0.032
   /ToleranceYellow [-0.02 0.02]
   /DotGain 13.3
   /ToleranceDotGain [2.0 -2.0]
   /Percentage 50.0
   /Diameter 5.0 mm
   /Type /Density
>> def
/Magenta50 <<
   /Screen (60 / cm)
   /Separation (Magenta)
   /DensityBlack 0.347
   /ToleranceBlack [-0.02 0.02]
   /DensityCyan 0.230
   /ToleranceCyan [-0.02 0.02]
   /DensityMagenta 0.413
   /ToleranceMagenta [-0.02 0.02]
   /DensityYellow 0.069
   /ToleranceYellow [-0.02 0.02]
   /DotGain 12.0
   /ToleranceDotGain [2.0 -2.0]
   /Percentage 50.0
   /Diameter 5.0 mm
   /Type /Density
>> def
/Yellow50 <<
   /Screen (60 / cm)
   /Separation (Yellow)
   /DensityBlack 0.357
   /ToleranceBlack [-0.02 0.02]
   /DensityCyan 0.157
   /ToleranceCyan [-0.02 0.02]
   /DensityMagenta 0.284
   /ToleranceMagenta [-0.02 0.02]
   /DensityYellow 0.357
   /ToleranceYellow [-0.02 0.02]
   /DotGain 5.7
   /ToleranceDotGain [2.0 -2.0]
   /Percentage 50.0
   /Diameter 5.0 mm
   /Type /Density
>> def
100.0 mm 15.5 mm Black50 CIP3PlaceMeasuringField
105.0 mm 15.5 mm Cyan50 CIP3PlaceMeasuringField
110.0 mm 15.5 mm Magenta50 CIP3PlaceMeasuringField
```

```
115.0 mm 15.5 mm Yellow50 CIP3PlaceMeasuringField
635.0 mm 689 mm 0 20 mm 5 mm
   [
     [ 0.0 mm 0 mm Black50 ]
     [ 5.0 mm 0 mm Cyan50 ]
     [ 10.0 mm 0 mm Magenta50 ]
     [ 15.0 mm 0 mm Yellow50 ]
   ]
   /UserControlStrip CIP3PlaceColorControlStrip
CIP3EndColorControl
```

## 3.9. Cutting Data

All necessary data required for cutting are compiled in the ***CutData*** structure.

Within the CIP3 format cutting data is described by nested blocks, where at the lowest level of hierarchy a folding procedure may be applied on a block. Since cutting is described here in a way that is as much as possible device independent, the cutting blocks specified in the CIP3 format do not directly imply a certain cutting sequence. This has to be determined by a specialized application, that also generates the cutting program, which can be loaded and executed on a cutting device.

Some information about the sheet which is also required for the cutting procedure is stored in attributes of the parent structures (e.g. in the *Sheet* structure or in the *Front* structure). These attributes (e.g. administration data) are inherited from the parent blocks. Therefore this information may not be included in the *CutData* structure. Although it is possible to overwrite these attributes by defining them in a cut block. This can be useful to specify different product identifications for the blocks arranged on a sheet.

#### Defining a cut block

It is possible to define a block which contains a matrix of elements of all the same size. There, the intermediate cut dimension is calculated from the information about element size, block size and number of elements in both directions. A cut block structure must be enclosed in a pair of ***CIP3BeginCutBlock*** and ***CIP3EndCutBlock*** commands. Cut block structures may be nested.

| Attribute Name | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3CutModel** | string | optional | Name of the cutting machine |

**Table 3-49: Attributes used in the CutData structure**

| Attribute Name | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3BlockTrf** | array of 6 numbers | required | Block transformation matrix: defines position and orientation of the block relative to the parent block or to the PostScript coordinate system (if the block is the root of the hierarchy of cut blocks; see chapter 3.1.3). |
| **CIP3BlockSize** | array of 2 numbers | required | Size of the block |
| **CIP3BlockElementSize** | array of 2 numbers | optional | Element dimension in X and Y direction |
| **CIP3BlockSubdivision** | array of 2 integers | optional | Number of elements in X and Y direction |
| **CIP3BlockType** | name | required | Block type: **/CutBlock** = block to be cut, **/SaveBlock** = protected block, cut only via outer contour, **/TempBlock** = auxiliary block, is not taken into account during cutting, **/MarkBlock** = contains no elements, only marks |
| **CIP3BlockElementType** | name | optional | Element type: **/CutElement** = cutting element, **/PunchElement** = punching element |
| **CIP3BlockName** | string | required | Name of the block. The name must be unique and must not contain a slash character (or the Unicode equivalent character, if coded in Unicode). |
| **CIP3BlockFoldingProcedure** | name | optional | Reference to folding procedure specified in the *FoldProcedures* section. If a block is subdivided into an array of elements, this folding procedure is applied to each element, otherwise it is applied to the whole block. |

**Table 3-50: Attributes used in the CutBlock structure**

### Description of cut marks

In addition to the definition of cut blocks it is possible to position cut marks on the sheet by using the ***CIP3PlaceCutMark*** command. After printing these marks can be used to adapt the theoretical block positions (as specified in the CIP3) to the real position of the corresponding blocks on the printed sheet.

| Parameter | Type | Meaning |
|---|---|---|
| ***position-x***<br>***position-y*** | number<br>number | Logical position of the cut mark. This coordinate does not always directly specify the position (e.g. the center) of the cut mark symbol. See Illustration 3-26: “The set of predefined cut marks” for the relation between the logical position and the position of the mark symbol. |
| ***mark-type*** | name | Mark type: **/CrossCutMark**, **/TopVerticalCutMark**, **/BottomVerticalCutMark**, **/LeftHorizontalCutMark**, **/RightHorizontalCutMark**, **/LowerLeftCutMark**, **/UpperLeftCutMark**, **/LowerRightCutMark**, **/UpperRightCutMark**. See Illustration 3-26: “The set of predefined cut marks” for the geometric shape of each predefined mark type. |

**Table 3-51: Parameter for the CIP3PlaceCutMark command**

| Symbol | Name | Position of symbol |
|---|---|---|
|  | CrossCutMark | Centered at logical position |
|  | TopVerticalCutMark | Slightly above logical position |
|  | BottomVerticalCutMark | Slightly below logical position |
|  | LeftHorizonalCutMark | Slightly to the left of logical position |
|  | RightHorizonalCutMark | Slightly to the right of logical position |
|  | LowerLeftCutMark | Corner at logical position |
|  | UpperLeftCutMark | Corner at logical position |
|  | LowerRightCutMark | Corner at logical position |
|  | UpperRightCutMark | Corner at logical position |

**Illustration 3-26: The set of predefined cut marks**

**Example 3-33: Block Definition**

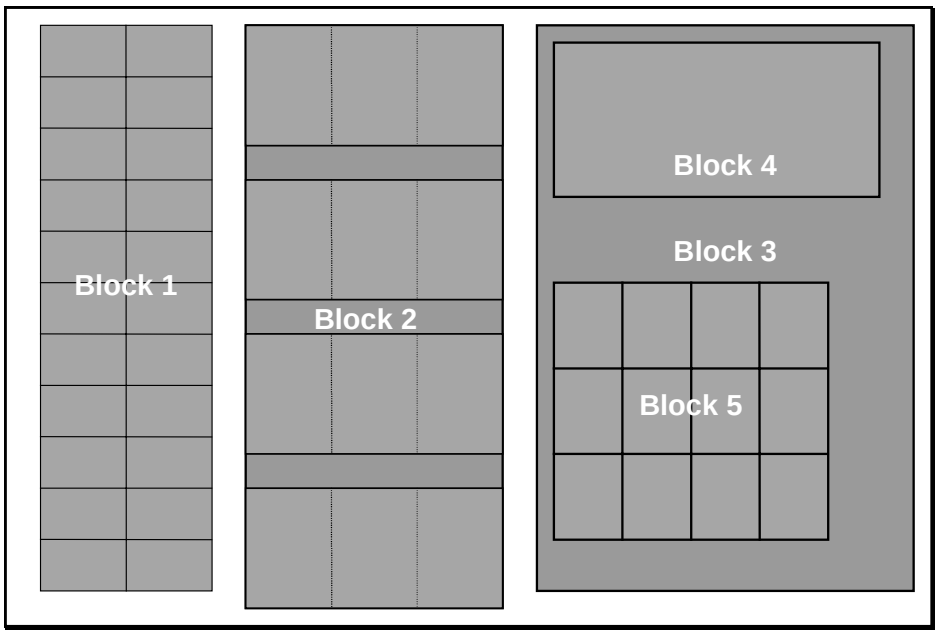



**Illustration 3-27: Example of a cutting arrangement**

```
CIP3BeginCutData
/CIP3CutModel (Model 175) def

CIP3BeginCutBlock
/CIP3BlockTrf         [1 0 0 1 4 cm  4 cm] def
/CIP3BlockSize        [20.2 cm 66.22 cm] def
/CIP3BlockElementSize [10.1 cm 6.02 cm] def
/CIP3BlockSubdivision [2 11] def
/CIP3BlockType        /CutBlock def
/CIP3BlockElementType /Unknown def
/CIP3BlockName        (Block 1) def
CIP3EndCutBlock

CIP3BeginCutBlock
/CIP3BlockTrf         [1 0 0 1 28.25 cm  1.5 cm] def
/CIP3BlockSize        [29.9 cm 68.7 cm] def
/CIP3BlockElementSize [29.9 cm 16.8 cm] def
/CIP3BlockSubdivision [1 4] def
/CIP3BlockType        /CutBlock def
/CIP3BlockElementType /Unknown def
/CIP3BlockName        (Block 2) def
/CIP3FoldingProcedure /DoubleFold def
CIP3EndCutBlock

CIP3BeginCutBlock
```

```
/CIP3BlockTrf         [1 0 0 1 63.0 cm  4.0 cm] def
/CIP3BlockSize        [29.9 cm 68.7 cm] def
/CIP3BlockType        /TempBlock def
/CIP3BlockName        (Block 3) def
CIP3BeginCutBlock
/CIP3BlockTrf         [1 0 0 1 2.0 cm  54.1 cm] def
/CIP3BlockSize        [38.8 cm 18.2 cm] def
/CIP3BlockType        /CutBlock def
/CIP3BlockName        (Block 4) def
CIP3EndCutBlock
CIP3BeginCutBlock
/CIP3BlockTrf         [1 0 0 1 4.0 cm  6.5 cm] def
/CIP3BlockSize        [32.4 cm 30.3 cm] def
/CIP3BlockElementSize [8.1 cm 10.1 cm] def
/CIP3BlockSubdivision [4 3] def
/CIP3BlockType        /CutBlock def
/CIP3BlockElementType /PunchElement def
/CIP3BlockName        (Block 5) def
CIP3EndCutBlock
CIP3EndCutBlock
CIP3EndCutData
```

**Example 3-34: Cut Marks**

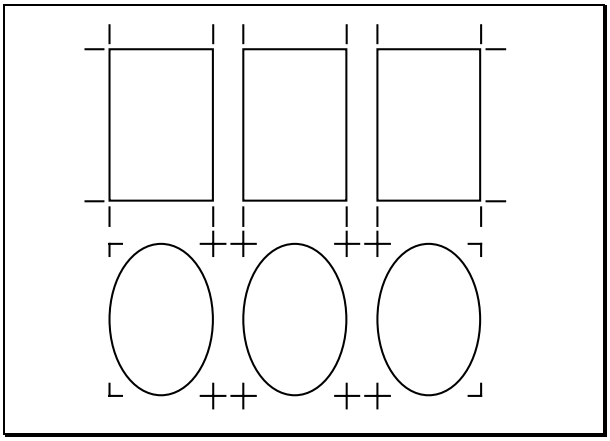


**Illustration 3-28: Example showing the use of cut marks**

```
15.1 cm  52.0 cm  /TopVerticalCutMark  CIP3PlaceCutMark
29.4 cm  52.0 cm  /TopVerticalCutMark CIP3PlaceCutMark
33.6 cm  52.0 cm  /TopVerticalCutMark CIP3PlaceCutMark
47.9 cm  52.0 cm  /TopVerticalCutMark CIP3PlaceCutMark
52.1 cm  52.0 cm  /TopVerticalCutMark CIP3PlaceCutMark
66.4 cm  52.0 cm  /TopVerticalCutMark CIP3PlaceCutMark
15.1 cm  52.0 cm  /LeftHorizontalCutMark CIP3PlaceCutMark
66.4 cm  52.0 cm  /RightHorizontalCutMark CIP3PlaceCutMark
15.1 cm  31.0 cm  /LeftHorizontalCutMark CIP3PlaceCutMark
66.4 cm  31.0 cm  /RightHorizontalCutMark CIP3PlaceCutMark
15.1 cm  31.0 cm  /BottomVerticalCutMark CIP3PlaceCutMark
29.4 cm  31.0 cm  /BottomVerticalCutMark CIP3PlaceCutMark
33.6 cm  31.0 cm  /BottomVerticalCutMark CIP3PlaceCutMark
47.9 cm  31.0 cm  /BottomVerticalCutMark CIP3PlaceCutMark
52.1 cm  31.0 cm  /BottomVerticalCutMark CIP3PlaceCutMark
66.4 cm  31.0 cm  /BottomVerticalCutMark CIP3PlaceCutMark
15.1 cm  26.0 cm  /UpperLeftCutMark CIP3PlaceCutMark
29.4 cm  26.0 cm  /CrossCutMark CIP3PlaceCutMark
33.6 cm  26.0 cm  /CrossCutMark CIP3PlaceCutMark
47.9 cm  26.0 cm  /CrossCutMark CIP3PlaceCutMark
52.1 cm  26.0 cm  /CrossCutMark CIP3PlaceCutMark
66.4 cm  26.0 cm  /UpperRightCutMark CIP3PlaceCutMark
15.1 cm   5.0 cm  /LowerLeftCutMark CIP3PlaceCutMark
29.4 cm   5.0 cm  /CrossCutMark CIP3PlaceCutMark
33.6 cm   5.0 cm  /CrossCutMark CIP3PlaceCutMark
47.9 cm   5.0 cm  /CrossCutMark CIP3PlaceCutMark
52.1 cm   5.0 cm  /CrossCutMark CIP3PlaceCutMark
66.4 cm   5.0 cm  /LowerRightCutMark CIP3PlaceCutMark
```

## 3.10. Folding Data

The creation of a folded product is described by a folding procedure. In the ***FoldProcedures*** section, a folding procedure can be defined. It is possible to specify more than one folding procedure in a CIP3 file. Each procedure is stored under a unique name by which it can be referenced in the *CutData* and *CutBlock* structures.

The cutting information contained in this section is only intended for the cutting procedures within the folding equipment.

#### Defining a folding procedure

A folding procedure is defined as a PostScript dictionary describing the folding parameters including the sequence of folding steps. At the moment the following five applications of a folding device are possible: folding, cutting, grooving, perforating and liming.

At the beginning of a folding procedure definition it is necessary to specify the size of the input sheet. If this size does not match the size of the corresponding cut block, all coordinates of the folding procedure are scaled respectively. This allows for the specification of a folding procedure that can be used in different sizes.

After each folding or cutting step of a folding procedure the origin of the coordinate system is moved to the lower left corner of the intermediate folding product.

| Key | Type | Necessary | Meaning |
|---|---|---|---|
| **CIP3FoldDescription** | string | optional | Description of the type of fold according to Bundesverband Druck (Federal Association of Printing) i.e. general description.<br>*Hint: Within a PostScript string a parenthesis „( )“ must be marked with a ´\´ in front of it.* |
| **CIP3FoldSheetIn** | array of 2 numbers | required | Input sheet format |
| **CIP3FoldProc** | array | required | Steps of folding procedure |

**Table 3-52: Data dictionary for folding procedure**

The specification of reference edges (**/Front**, **/Rear**, **/Left** and **/Right**) for the description of an operation (e.g. for the positioning of a tool) is done by means of determined names. With regard to upper and lower case they have to be written exactly as in the following Illustration 3-29.

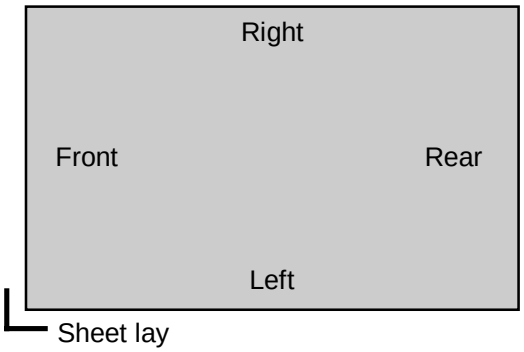



**Illustration 3-29: Names of the reference edges of a sheet**

The individual steps of a folding procedure must be given in the sequence in which they should be carried out. Each step requires the specification of some parameter values and is concluded with the indication of the application name (**Fold**, **Cut**, **Groove**, **Perforate** or **Lime**). The direction of folding is defined by means of a name (**/Up** or **/Down**).

| Parameter | Type | Meaning |
|---|---|---|
| ***travel*** | number | Distance of the reference edge (*from*) |
| ***from*** | name | Edge from where it is folded: **/Front** = from the front, **/Left** = from the left |
| ***to*** | name | Direction in which it is folded: **/Up** = upwards, **/Down** = downwards. |
| ***function*** | name | Function of the operation: **Fold** = folding |

**Table 3-53: Description of a folding operation**

For indicating the direction in which a tool (for cutting, grooving, perforating, or liming) processes the sheet, the following names have been defined: **/Bottom** and **/Top**.

| Parameter | Type | Meaning |
|---|---|---|
| ***start-position*** | 2 numbers | Starting position of the tool |
| ***working-path*** | 2 numbers | Relative working path of the tool. Since the tools can only work parallel to the edges, one coordinate must be zero. |
| ***working-direction*** | name | Direction from which the tool is working:<br>**/Top** = from above, **/Bottom** = from below |
| ***function*** | name | Function of the operation:<br>**Cut** = cutting, **Groove** = grooving, **Perforate** = perforating, **Lime** = liming |

**Table 3-54: Description of an operation for cutting, grooving, perforating or liming**

**Example 3-35: Folding Procedure**

*Note: The following example refers to the third example in the description of the PrePress Interface of Stahl (version of 22.8.94, section 3.5.3, pages 15-17).*

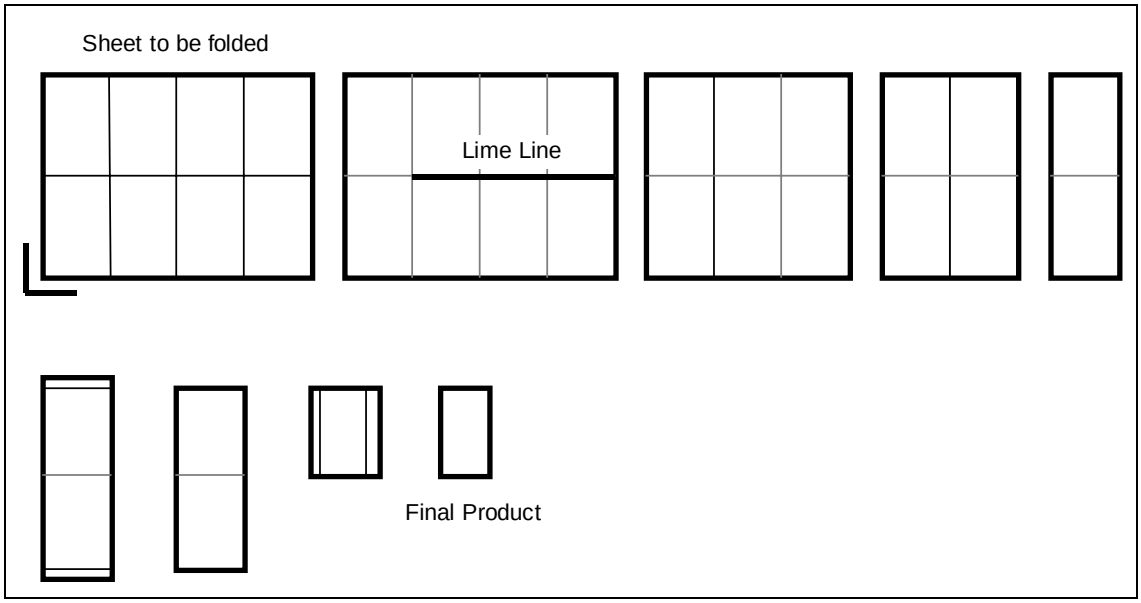



**Illustration 3-30: Example of a folded product**

```
CIP3BeginFoldProcedures
/A4-16  <<
            /CIP3FoldDescription (F16 / 3W \(1/4\) + 1 ML)
            /CIP3FoldSheetIn [880 mm 610 mm]
            /CIP3FoldProc
            [
               220 mm 305 mm 660 mm 0 /Top Lime
               220 mm /Front /Up Fold                % origin moves right
               220 mm /Front /Up Fold                % origin moves right
               220 mm /Front /Up Fold                % origin moves right
                 0 mm   8 mm 220 mm 0 /Top Cut       % origin moves down
                 0 mm 594 mm 220 mm 0 /Top Cut
               297 mm /Left /Up Fold                 % origin moves up
                 5 mm   0 mm 0 297 mm /Top Cut       % origin moves right
               210 mm   0 mm 0 297 mm /Top Cut
            ]
         >> def
CIP3EndFoldProcedures
```

## 3.11. Comments and Annotations

Comments and annotations are treated as content. A comment is specified by typing the comment text and the ***CIP3Comment*** command. It is not printed. On the contrary, an annotation is a printed comment, which is specified by the ***CIP3Annotation*** command.

| Parameter | Type | Meaning |
|---|---|---|
| ***text*** | string | Comment string. |

**Table 3-55: Parameter for the CIP3Comment command**

| Parameter | Type | Meaning |
|---|---|---|
| ***position-x*** | number | Starting position of text baseline in X-direction. |
| ***position-y*** | number | Starting position of text baseline in Y-direction. |
| ***text*** | string | Annotation text |
| ***font-name*** | name | Name of font to be used for printing |
| ***font-size*** | number | Font size |

**Table 3-56: Parameters for the CIP3Annotation command**

**Example 3-36: Comments and Annotations**

```
(This is a not printed comment) CIP3Comment
10 cm 12.3 cm (This is a printed comment) /Helvetica 12 CIP3Annotation
```

## 3.12. Private Data

For many applications it is comfortable to store additional, application specific data within the CIP3 PPF file. This can be done by use of a ***PrivateData*** structure. This structure can be inserted within any other structure. Even nesting of *PrivateData* structures is allowed. *PrivateData* structures have to be named using the following rule:

- the name of the *PrivateData* structure must start with a company specific prefix containing at least three characters. It is not allowed to use names starting with the "CIP3" prefix. The company specific prefix should be registered by the CIP3 group.

Within a *PrivateData* structure arbitrary definitions can be made. Each attribute can be defined by use of the "def" operator.

```
/attribute-name value def
```

The attribute name must be unique within one *PrivateData* structure. Although it is not required to use a company specific prefix for attribute name, this is regarded as good coding technique. The value has to be provided as a PostScript object (see chapter 3.1.2 for a complete list of supported data types). Arrays and dictionaries may be used to group information.

**Example 3-37: Private Data**

```
/FhGPrivate CIP3BeginPrivate
/FhGPrivValue 42 def
/FhGPrivDictionary <</Company (Fraunhofer)
                     /Date (07.03.96)
                     /Array [2.4 5.4 (Text)]
                   >> def
CIP3EndPrivate
```

## 3.13. Private Content

Since binary data consisting of more than 65535 bytes cannot be represented as a PostScript string, the concept of private content has been introduced. With this method the amount of data that can be stored in one single attribute is only limited by the available memory. The length of the data in bytes must be provided in order to allow the data being skipped by positioning the file pointer.

The private content data may contain arbitrary binary data without any restriction. The data bytes start after one white space character following the keyword ***CIP3PrivateContent***. The combination of carriage return followed by line-feed is regarded as one white space character (see chapter 3.8.1 of the PostScript Language Reference Manual [PostScript] for a more detailed description). If carriage return only is used as EOL convention (as on Macintosh computers) and the first character of the private content might be line-feed, it is recommended to use the space character to separate the *CIP3PrivateContent* keyword from the private content data.

```
name length CIP3PrivateContent
< ... length bytes of data ... >
```

**Example 3-38: Private Content**

```
...

/FhGPrivateDataElem 24 CIP3PrivateContent
123456789012345678901234

...
```

# Appendix A. Changes between different versions

### A.1 Changes from CIP3 PPF version 2.1 to version 3.0

The following changes to the CIP3 PPF specification document have been made (the chapter numbers given in parentheses refer to the chapter of the specification document version 3.0):

- **Introduction and Overview** (chapter 1 and chapter 2)
  The introduction and overview chapters have been revised to reflect the addition of the product definition.
- **Encoding in PostScript** (chapter 2.4)
  It is now stated clearly that attributes must occur in a structure before the first content element and before the next “CIP3Begin...”.
- **version number changed** (chapter 3.1.1)
  The version number has been changed from 2.1 to 3.0.
- **optional header line added** (chapter 3.1.1)
  A third, optional, header line has been added to the CIP3 PPF file header to increase the probability of a CIP3 PPF file being identified as a binary file.
- **chapter "Syntax" renamed and reorganized** (chapter 3.1.2)
  The chapter "Syntax" has been renamed to "Syntax and Data Types". In addition table 3-1 (of specification version 2.1) has been divided into several chapters, one for each data type.
- **recommendation for using 7-Bit ASCII removed** (chapter 3.1.2)
  Since CIP3 PPF files are likely to contain binary data (e.g. for the preview images or in Unicode encoded strings), the recommendation for using only 7-Bit ASCII codes has been removed. Instead an optional third header line has been added that identifies the CIP3 PPF file as a binary file (see above).
- **number data type added** (chapter 3.1.2.4)
  A new data type called "number" has been added. It represents a number, which can be coded as a real or as an integer. All occurrences of the "float" data type have been replaced by the new number data type.
- **Unicode support added** (chapter 3.1.2.6)
  Any string within the CIP3 PPF may now be encoded in Unicode. Strings encoded in Unicode can be recognized by the Unicode Byte Order marker.
- **structure types changed** (chapter 3.1.4)
  The structure types of the Front and Back structures have both been changed to the new structure type "Surface". Although, this will not cause any change in how to code a CIP3 PPF file.
- **definition of valid CIP3 PPF structure tree changed** (chapter 3.1.4)
  The definition of a valid structure tree (see Table 3-4: “Rules defining all valid CIP3 PPF structure trees”) has been changed to reflect the change mentioned above. The rules now include the private data structure.
- **"name" parameter of CIP3PPFDirEntry clarified** (chapter 3.2)
  It is now explicitly required that the "name" parameter must be copied from the CIP3AdmSheetName attribute as defined in the PPF subfile and that it must be a unique sheet name (at least within the whole CIP3 PPF file). Furthermore it must not contain a slash character (or the Unicode equivalent character, if coded in Unicode).
- **product definition added** (chapter 3.3)
  This is the most important difference between version 2.1 and version 3.0. Within the CIP3 PPF it is now possible to specify a complete product (e.g. a brochure).

- **explanation of administration data attributes improved** (chapter 3.4)
  The "Meaning" field of the following attributes has been changed:

| | |
|---|---|
| CIP3AdmCreationTime | Two alternative string formats are now recommended. |
| CIP3AdmSheetName | Changed according to the change of the CIP3PPFDirEntry parameter "name". |
| CIP3AdmSeparationNames | The sequence of separation names must match the sequence of the separations as stored within the CIP3 PPF file. For standard process colors the strings (Cyan), (Magenta), (Yellow), and (Black) must be used. |
| CIP3AdmInkInfo | The sequence of the ink information strings must match the sequence of the separations as specified by the CIP3AdmSeparationNames attribute. |
| CIP3AdmInkColors | The sequence of the ink color specification arrays must match the sequence of the separations as specified by the CIP3AdmSeparationNames attribute. |
| CIP3AdmPSExtent | The PS extent is now explicitly defined as the exact size of the preview images. |
| CIP3AdmFilmTrf | The default value is now explicitly specified. |
| CIP3AdmFilmExtent | The inheritance is now explicitly specified. |
| CIP3AdmPlateTrf | The default value is now explicitly specified. |
| CIP3AdmPlateExtent | The inheritance is now explicitly specified. |
| CIP3AdmPressTrf | The default value is now explicitly specified. |
| CIP3AdmPressExtent | The inheritance is now explicitly specified. |
| CIP3AdmPaperTrf | The default value is now explicitly specified. |
| CIP3AdmPaperExtent | The inheritance is now explicitly specified. |

- **preview image** (chapter 3.5)
  It is now explicitly specified that the CIP3AdmPSExtent must match the size of the preview image. A deviation of ±1 pixel is allowed.
- **composite preview image** (chapter 3.5.1)
  The meaning of 8-bit color components is now explicitly specified.
- **preview image with separations** (chapter 3.5.2 )
  The meaning of 1-bit and 8-bit color components is now explicitly specified.
- **characteristic curves for transfer** (chapter 3.6)
  The identity transfer curve is explicitly given. Furthermore an example has been added explaining how to specify and how to use the characteristic curves for transfer.
- **density measuring fields** (chapter 3.8)
  The type of the /Separation entry within the data dictionary of a density measuring field has been corrected to "name".
  Furthermore the example has been corrected, which was using the PostScript operators "begin", "end" and "dict" for the creation of dictionaries, although these operators are not supported since CIP3 PPF version 2.1. Instead, dictionaries are now created by using "<<" and ">>".

- **comments and annotations** (chapter 3.11)
  Two tables have been added to show the parameters of the CIP3Comment and CIP3Annotation commands.
- **list of registered names** (appendix D)
  Two tables have been added that list all registered names of the CIP3 Print Production Format.

### A.2 Changes from CIP3 PPF version 2.0 to version 2.1

The following changes to the CIP3 PPF specification document have been made (the chapter numbers given in parentheses refer to chapters or illustrations of the specification document version 2.1):

- **version number changed** (chapter 3.1.1)
  The version number has been changed from 2.0 to 2.1.
- **syntactical structure of CIP3 file** (illustration 3-1)
  The illustration has been adapted to show the PPFDirectory and the possibility of describing more than one sheet within one CIP3 PPF file.
- **explanation of how to code objects** (chapter 3.1.2)
  A description has been added of how to code objects of type boolean, integer, real, string, name, array, and dict. In addition the coding of PostScript comments has been explained.
- **specification of syntax limitations** (chapter 3.1.2)
  In order to further reduce the complexity of a CIP3 PPF file the following PostScript operators have been removed from the list of supported operators:
  `begin`, `end`, `dict`, `{`, `}`.
- **What is a valid CIP3 PPF file?** (chapter 3.1.5)
  A chapter explaining what a valid CIP3 PPF file must consist of has been added.
- **PPF directory** (chapter 3.2)
  A PPF directory was added to support multi sheet PPF files.
- **administration data** (chapter 3.3)
  The CIP3AdmSheetLay attribute has been changed from required to optional and a short explanation has been added. A new optional attribute CIP3AdmTypeOfScreen has been has been added. A set of attributes for web presses has been added.
- **structure names** (chapter 3.3 and chapter 3.9)
  No slash characters may be used in structure names. Therefore the separation names (stored in the CIP3AdmSeparationNames array) and the names of a cut blocks (stored in CIP3BlockName) must not contain slash characters.
- **preview image data** (chapter 3.5)
  A new optional attribute specifying the size of the encoded and compressed image data has been added. The fax compression mode has been corrected (changed from CCITTDecode to CCITTFaxDecode).
- **using image encoding and compression** (chapter 3.5.5)
  A new chapter has been added with hints for using PostScript encoding and compression filters to code preview image data.
- **characteristic curves of transfer** (chapter 3.6)
  A diagram explaining the semantic of the curves of transfer has been added. It is now allowed to store separation specific curves of transfer in the PPF file.
- **color and ink control** (chapter 3.8)
  An optional entries specifying the percentage values for each separation, the density filter standard, and the shape of the screening dots has been added to the dictionary for color measuring fields.
- **cutting data** (chapter 3.9)
  The attributes of a CutData structure have been clearly separated from the attributes of a CutBlock structure. The data type used for CIP3BlockTrf has changed from "matrix" to "array of 6 floats".

- **folding data** (chapter 3.10)
  Instead of a PostScript procedure (with "{" and "}" brackets) a dictionary notation (with "<<" and ">>" brackets) is used to code a folding procedure. The sequence of working steps is then coded in an array stored in the CIP3FoldProc attribute
- **private content** (chapter 3.13)
  A means of storing large blocks of private data has been established by adding the concept of private content.
- **dictionary elements**
  All dictionary elements have been explicitly marked as optional or required.
- **list of illustrations (appendix E)**
  A list of illustrations has been added.
- **list of tables (appendix F)**
  A list of tables has been added.

### A.3 Changes from CIP3 PPF version 1.0 to version 2.0

The following changes to the CIP3 PPF specification document have been made (the chapter numbers given in parentheses refer to the chapter of the specification document version 2.0):

- **version number changed** (chapter 3.1.1)
  The version number has been changed from 1.0 to 2.0.
- **prologue removed** (chapter 3.1.1)
  In order to reduce the complexity of a CIP3 PPF file it is no longer possible to make user specific definition within a prologue.
- **specification of syntax limitations** (chapter 3.1.2.)
  In order to reduce the complexity of a CIP3 PPF file some syntax limitations have been defined. In addition a complete list of all PostScript operators that may be used in a CIP3 PPF file has been added.
- **specification of default value for extent and transformation attributes** (chapter 3.1.3)
  The default values for an unspecified transformation matrix (CIP3AdmFilmTrf, CIP3AdmPlateTrf, CIP3AdmPressTrf, CIP3AdmPaperTrf) and an unspecified extent (CIP3AdmFilmExtent, CIP3AdmPlateExtent, CIP3AdmPressExtent, CIP3AdmPaperExtent) have been defined (see chapter 3.2.: administration data).
- **change of necessary flag of two administration data attributes** (chapter 3.2.)
  The CIP3AdmSeparationNames and the CIP3AdmPSExtent attributes have been changed from optional to required.
- **changes to the preview image** (chapter 3.3)
  In order to reduce to size of a CIP3 PPF file encoding and compression techniques can be applied to preview images. Since the use of LZW compression requires a license fee paid to Unisys, LZW is excluded from the list of supported compression techniques.
  The coding of preview images has been changed from an OPI-like style to the definition of attributes similar to the ones used for administration data. Due to the problems related with links to external files, the OPI-like file references are no longer supported.
  In addition to continuous tone images the new version also allows bitmap images (1 bit per pixel) to be used as preview images. Of course these images must be provided in a higher resolution.
- **specification of register marks associated with a set of separations** (chapter 3.5)
  In addition to their position on the sheet register marks can now be associated with a set of separations. By default a register mark is associated with each separation of the job (as in version 1.0).
- **private data added** (chapter 3.10.)
  To allow for the definition of private data within a CIP3 PPF file, a separate structure has been defined.
- **changes to the list of references** (appendix A)
  The references to the OPI and TIFF specifications have been removed. In addition the reference to the Document Structuring Conventions Specification has been removed, since it is part of the PostScript Language Reference Manual [PostScript].
- **folding data now in separate structure** (chapter 3.8.)
  The definition of fold procedures has been moved from the beginning of the file (just after the removed prolog) to a separate structure.

# Appendix B. Example of a CIP3 PPF file

*Note: The example shown below is collection of most of the examples shown in previous sections of this specification. Its logical structure is shown in Illustration 2-1 on page 7. The sheet described with this CIP3 PPF file comprises a front side with four colors (CMYK) and a back side with two colors (Cyan and Black).*

**Example 3-39: Complete CIP3 PPF File**

```
%!PS-Adobe-3.0
%%CIP3-File Version 3.0
%ÔÒ¤Ë
   < ... the PPF Directory would be inserted here ... >
   < ... the Product Definition would be inserted here ... >

CIP3BeginSheet
(Sheet structure of CIP3 example) CIP3Comment
/CIP3AdmJobName (TestJob) def
/CIP3AdmMake (Prepress Company) def
/CIP3AdmModel (PC210) def
/CIP3AdmSoftware (The Imposition Program) def
/CIP3AdmCreationTime (Thu Jan 28 12:25:12 1998) def
/CIP3AdmArtist (Stefan Daun) def
/CIP3AdmCopyright (Copyright by Fraunhofer-IGD, 1995) def
/CIP3AdmPSExtent [100 cm 70 cm] def
/CIP3AdmSheetLay /Left def
/CIP3AdmPrintVolume 120000 def
/CIP3AdmPaperGrammage 130 def

(Transfer data is valid for both front and back) CIP3Comment
/CIP3TransferFilmCurveData  [ 0.0   0.0     0.2   0.3     0.35   0.5
                              0.5   0.65    0.7   0.8     1.0    1.0  ] def
/CIP3TransferPlateCurveData [ 0.0   0.0     0.3   0.25    0.475  0.4
                              0.6   0.525   0.75  0.7     1.0    1.0  ] def

CIP3BeginFront
(Front page structure of CIP3 example) CIP3Comment
(Different number of colors for front and back side) CIP3Comment
/CIP3AdmSeparationNames [(Cyan) (Magenta) (Yellow) (Black)] def

CIP3BeginPreviewImage
/CIP3PreviewImageWidth 2000 def
/CIP3PreviewImageHeight 1400 def
/CIP3PreviewImageBitsPerComp 8 def
/CIP3PreviewImageComponents 4 def
/CIP3PreviewImageMatrix [2000 0 0 -1400 0 1400] def
/CIP3PreviewImageResolution [ 50.8 50.8 ] def
/CIP3PreviewImageEncoding /ASCIIHexDecode def
/CIP3PreviewImageCompression /DCTDecode def
```

```
/CIP3PreviewImageFilterDict <<>> def
CIP3PreviewImage
... <image data>
CIP3EndPreviewImage

CIP3BeginRegisterMarks
30.0 30.0 0 /regm1 CIP3PlaceRegisterMark
625.0 30.0 0 /regm1 CIP3PlaceRegisterMark
625.0 872.0 0 /regm1 CIP3PlaceRegisterMark
30.0 872.0 0 /regm1 CIP3PlaceRegisterMark

/FhGPrivate CIP3BeginPrivate
/FhGPrivValue 42 def
/FhGPrivDictionary << /Company (Fraunhofer)
                      /Date (07.03.96)
                      /Array [ 2.4 5.4 (Text)]
                   >> def
CIP3EndPrivate

CIP3EndRegisterMarks

CIP3BeginColorControl
/Black50 <<
   /Screen (60 / cm)
   /Separation (Black)
   /DensityBlack 0.331
   /ToleranceBlack [-0.02 0.02]
   /DensityCyan 0.293
   /ToleranceCyan [-0.02 0.02]
   /DensityMagenta 0.313
   /ToleranceMagenta [-0.02 0.02]
   /DensityYellow 0.054
   /ToleranceYellow [-0.02 0.02]
   /DotGain 15.2
   /ToleranceDotGain [2.0 -2.0]
   /Percentage 50.0
   /Diameter 5.0 mm
   /Type /Density
>> def
/Cyan50 <<
   /Screen (60 / cm)
   /Separation (Cyan)
   /DensityBlack 0.338
   /ToleranceBlack [-0.02 0.02]
   /DensityCyan 0.446
   /ToleranceCyan [-0.02 0.02]
   /DensityMagenta 0.043
   /ToleranceMagenta [-0.02 0.02]
   /DensityYellow 0.032
```

```
   /ToleranceYellow [-0.02 0.02]
   /DotGain 13.3
   /ToleranceDotGain [2.0 -2.0]
   /Percentage 50.0
   /Diameter 5.0 mm
   /Type /Density
>> def
/Magenta50 <<
   /Screen (60 / cm)
   /Separation (Magenta)
   /DensityBlack 0.347
   /ToleranceBlack [-0.02 0.02]
   /DensityCyan 0.230
   /ToleranceCyan [-0.02 0.02]
   /DensityMagenta 0.413
   /ToleranceMagenta [-0.02 0.02]
   /DensityYellow 0.069
   /ToleranceYellow [-0.02 0.02]
   /DotGain 12.0
   /ToleranceDotGain [2.0 -2.0]
   /Percentage 50.0
   /Diameter 5.0 mm
   /Type /Density
>> def
/Yellow50<<
   /Screen (60 / cm)
   /Separation (Yellow)
   /DensityBlack 0.357
   /ToleranceBlack [-0.02 0.02]
   /DensityCyan 0.157
   /ToleranceCyan [-0.02 0.02]
   /DensityMagenta 0.284
   /ToleranceMagenta [-0.02 0.02]
   /DensityYellow 0.357
   /ToleranceYellow [-0.02 0.02]
   /DotGain 5.7
   /ToleranceDotGain [2.0 -2.0]
   /Percentage 50.0
   /Diameter 5.0 mm
   /Type /Density
>> def
100.0 mm 15.5 mm Black50 CIP3PlaceMeasuringField
105.0 mm 15.5 mm Cyan50 CIP3PlaceMeasuringField
110.0 mm 15.5 mm Magenta50 CIP3PlaceMeasuringField
115.0 mm 15.5 mm Yellow50 CIP3PlaceMeasuringField
635.0 mm 689 mm 0 20 mm 5 mm
   [
     [ 0.0 mm 0 mm Black50 ]
     [ 5.0 mm 0 mm Cyan50 ]
```

```
     [ 10.0 mm 0 mm Magenta50 ]
     [ 15.0 mm 0 mm Yellow50 ]
   ]
   /UserControlStrip CIP3PlaceColorControlStrip
CIP3EndColorControl

CIP3EndFront

CIP3BeginBack
(Back page structure of CIP3 example) CIP3Comment
/CIP3AdmSeparationNames [(Cyan) (Black)] def

CIP3BeginPreviewImage
CIP3BeginSeparation
/CIP3PreviewImageWidth 2000 def
/CIP3PreviewImageHeight 1400 def
/CIP3PreviewImageBitsPerComp 8 def
/CIP3PreviewImageComponents 1 def
/CIP3PreviewImageMatrix [2000 0 0 1400 0 0] def
/CIP3PreviewImageResolution [ 50.8 50.8 ] def
/CIP3PreviewImageEncoding /ASCII85Decode def
/CIP3PreviewImageCompression /RunLengthDecode def
CIP3PreviewImage <... runlength compressed and ASCII85 encoded
                     image data of Cyan separation ...>
CIP3EndSeparation
CIP3BeginSeparation
/CIP3PreviewImageWidth 2000 def
/CIP3PreviewImageHeight 1400 def
/CIP3PreviewImageBitsPerComp 8 def
/CIP3PreviewImageComponents 1 def
/CIP3PreviewImageMatrix [2000 0 0 1400 0 0] def
/CIP3PreviewImageResolution [ 50.8 50.8 ] def
/CIP3PreviewImageEncoding /Binary def
/CIP3PreviewImageCompression /None def
CIP3PreviewImage <... raw image data of Black separation; should be separated
             from CIP3PreviewImage keyword by only one single space character ...>
CIP3EndSeparation
CIP3EndPreviewImage

CIP3BeginRegisterMarks
30.0 30.0 0 /regm1 CIP3PlaceRegisterMark
625.0 30.0 0 /regm1 CIP3PlaceRegisterMark
625.0 872.0 0 /regm1 CIP3PlaceRegisterMark
30.0 872.0 0 /regm1 CIP3PlaceRegisterMark
CIP3EndRegisterMarks

CIP3EndBack
CIP3EndSheet
%%CIP3EndOfFile
```

# Appendix C. References

[BVD] Bundesverband Druck: Handbuch zur Anwendung einheitlicher Falzbezeichnungen.- Falzartenkatalog, 1990.

[CIE1] ISO 10526: CIE Standard Colorimetric Illuminance.

[CIE2] ISO 10527: CIE Standard Colorimetric Observers.

[DSC] Adobe Systems Inc.: PostScript Language Document Structuring Conventions Specification - Version 3.0 (September 25, 1992).

[PDF] Adobe Systems Inc.: Portable Document Format Reference Manual - Version 1.2 (November 27, 1996)
`http://www.adobe.com/supportservice/devrelations/PDFS/TN/PDFSPEC.PDF`

[PostScript] Adobe Systems Inc.: PostScript Language Reference Manual. Reading Mass.: Addison-Wesley, 1990, 2nd Edition.

# Appendix D. List of Registered Names

This appendix contains two tables that list all registered names of the CIP3 Print Production Format. The first table shows all registered names in alphabetic order, while the second table shows the attributes, dict entries and command parameters that are using these registered names.

Since it is much likely that that more names will be registered, an updated version of these tables is available on the CIP3 WWW Server at the following address:

`http://www.cip3.org/documents/technical_info`

| **Registered Name** | **Name Used In** |
|---|---|
| /AdhesiveBinding | CIP3ProductOperation |
| /ASCII85Decode | CIP3PreviewImageEncoding |
| /ASCIIHexDecode | CIP3PreviewImageEncoding |
| /BackPreparation | ProcessType |
| /Binary | CIP3PreviewImageEncoding |
| /Block | SourceType |
| /Bottom | *working-direction* |
| /BottomVerticalCutMark | *cut-mark-type* |
| /Butted | StapleShape |
| /Calico | LiningMaterial |
| /CCITTFaxDecode | CIP3PreviewImageCompression |
| /CIELAB | Type |
| /ClinchOut | StapleShape |
| /ColdGlue | GlueType |
| /Collecting | CIP3ProductOperation |
| /CombinedStaggered | SewingPattern |
| /Cotton | CastingMaterial |
| /Cotton | CoreMaterial |
| /Cotton | ThreadMaterial |
| /CrepePaper | LiningMaterial |
| /CrossCutMark | *cut-mark-type* |
| /Crown | StapleShape |
| /Cut | Function |
| /Cut | *function* |
| /CutBlock | CIP3BlockType |
| /CutElement | CIP3BlockElementType |
| /D50 | Light |
| /D65 | Light |
| /DCTDecode | CIP3PreviewImageCompression |
| /Density | Type |
| /Down | *to-direction* |
| /EndSheetGluing | CIP3ProductOperation |
| /ExternalProduct | SourceType |
| /Eyelet | StapleShape |
| /Fold | *function* |
| /Folded | CIP3AdmPaperDestination |
| /FromInside | ScoringSide |
| /FromOutside | ScoringSide |

| Registered Name | Name Used In |
|---|---|
| /Front | *from-edge* |
| /Gathering | CIP3ProductOperation |
| /Gauze | LiningMaterial |
| /Glue | Function |
| /GlueApplication | ProcessType |
| /GluingIn | CIP3ProductOperation |
| /Groove | *function* |
| /Hotmelt | GlueType |
| /Left | CIP3AdmSheetLay |
| /Left | *from-edge* |
| /Left | ReferenceEdge |
| /LeftHorizontalCutMark | *cut-mark-type* |
| /Lime | *function* |
| /Lining | ProcessType |
| /LowerLeftCutMark | *cut-mark-type* |
| /LowerRightCutMark | *cut-mark-type* |
| /MarkBlock | CIP3BlockType |
| /None | CIP3PreviewImageCompression |
| /Normal | SewingPattern |
| /Nylon | CastingMaterial |
| /Nylon | CoreMaterial |
| /Nylon | ThreadMaterial |
| /Overlap | StapleShape |
| /PaperlinedMules | LiningMaterial |
| /PartialProduct | SourceType |
| /Perforate | Function |
| /Perforate | *function* |
| /Polyester | CoreMaterial |
| /Polyester | ThreadMaterial |
| /PunchElement | CIP3BlockElementType |
| /PUR | GlueType |
| /Reel | CIP3AdmPaperDestination |
| /Reel | CIP3AdmPaperSource |
| /Right | CIP3AdmSheetLay |
| /Right | ReferenceEdge |
| /RightHorizontalCutMark | *cut-mark-type* |
| /RunLengthDecode | CIP3PreviewImageCompression |
| /SaddleStitching | CIP3ProductOperation |
| /SaveBlock | CIP3BlockType |
| /Sheet | CIP3AdmPaperDestination |
| /Sheet | CIP3AdmPaperSource |
| /Sheet | SourceType |
| /SideGluingBack | GluingTechnique |
| /SideGluingFront | GluingTechnique |
| /SideSewing | CIP3ProductOperation |
| /SpineGluing | GluingTechnique |
| /Staggered | SewingPattern |

| Registered Name | Name Used In |
|---|---|
| /Stitch | Function |
| /Stitching | CIP3ProductOperation |
| /TempBlock | CIP3BlockType |
| /ThreadSewing | CIP3ProductOperation |
| /Top | *working-direction* |
| /TopVerticalCutMark | *cut-mark-type* |
| /Trimming | CIP3ProductOperation |
| /Up | *to-direction* |
| /UpperLeftCutMark | *cut-mark-type* |
| /UpperRightCutMark | *cut-mark-type* |
| CoverApplication | ProcessType |
| Polyester | CastingMaterial |

**Table D-1: Registered names**

| Attribute, Dict Entry, or Command Parameter | Type | Meaning | Registered Names |
|---|---|---|---|
| CastingMaterial | dict entry in operation specific dictionary for ThreadSewing operation | Casting material of thread | /Cotton<br>/Nylon<br>/Polyester |
| CIP3AdmPaperDestination | attribute | Output destination of paper | /Reel<br>/Sheet<br>/Folded |
| CIP3AdmPaperSource | attribute | Input source of paper | /Reel<br>/Sheet |
| CIP3AdmSheetLay | attribute | Guiding edge of the paper in the press seen in the direction of paper flow | /Left<br>/Right |
| CIP3BlockElementType | attribute | Element type | /CutElement<br>/PunchElement |
| CIP3BlockFoldingProcedure | attribute | Reference to folding procedure specified in FoldProcedures section. | *no predefined names* |
| CIP3BlockType | attribute | Block type | /CutBlock<br>/SaveBlock<br>/TempBlock<br>/MarkBlock |

| Attribute, Dict Entry, or Command Parameter | Type | Meaning | Registered Names |
|---|---|---|---|
| CIP3PreviewImageCompression | attribute | Type of preview image compression | /None<br>/RunLengthDecode<br>/DCTDecode<br>/CCITTFaxDecode |
| CIP3PreviewImageEncoding | attribute | Type of preview image encoding | /Binary<br>/ASCIIHexDecode<br>/ASCII85Decode |
| CIP3ProductOperation | dict entry in product definition dictionary | Type of finishing operation | /Collecting<br>/Gathering<br>/EndSheetGluing<br>/ThreadSewing<br>/SaddleStitching<br>/Stitching<br>/SideSewing<br>/AdhesiveBinding<br>/Trimming<br>/GluingIn |
| CoreMaterial | dict entry in operation specific dictionary for ThreadSewing operation | Core material of thread | /Cotton<br>/Nylon<br>/Polyester |
| *cut-mark-type* | parameter of CIP3PlaceCutMark command | Mark type | /CrossCutMark<br>/TopVerticalCutMark<br>/BottomVerticalCutMark<br>/LeftHorizontalCutMark<br>/RightHorizontalCutMark<br>/LowerLeftCutMark<br>/UpperLeftCutMark<br>/LowerRightCutMark<br>/UpperRightCutMark |
| *font-name* | parameter of CIP3Annotation command | Font name | *not registered within CIP3* |
| *from-edge* | parameter of folding operation | Edge from where it is folded | /Front<br>/Left |
| Function | dict entry in data dictionary for longitudinal and cross applications | Type of function | /Cut<br>/Perforate<br>/Glue<br>/Stitch |

| Attribute, Dict Entry, or Command Parameter | Type | Meaning | Registered Names |
|---|---|---|---|
| *function* | parameter of one step in folding operation | Function of the operation | /Fold<br>/Cut<br>/Groove<br>/Perforate<br>/Lime |
| GlueType | dict entry in GlueLine dictionary | Glue type | /ColdGlue<br>/Hotmelt<br>/PUR |
| GluingTechnique | dict entry in dictionary for Glue Application | Type or technique of gluing application | /SpineGluing<br>/SideGluingFront<br>/SideGluingBack |
| Light | dict entry in data dictionary for color measuring field | Type of light | /D50<br>/D65 |
| LiningMaterial | dict entry in dictionary for Lining Process | Lining material | /Gauze<br>/Calico<br>/PaperlinedMules<br>/CrepePaper |
| *name-of-control-strip* | parameter of CIP3PlaceColorControlStrip | name of control strip | *not yet defined* |
| ProcessType | dict entry in dictionary for the following subprocesses of the Adhesive Binding process: Back Preparation, Glue Application, Lining, Cover Application | identifies the type of the subprocess | /BackPreparation<br>/GlueApplication<br>/Lining<br>CoverApplication |
| ReferenceEdge | dict entry in data dictionary for longitudinal and cross applications | Reference edge seen in direction of web travel | /Left<br>/Right |
| *register-mark-type* | parameter of CIP3PlaceRegisterMark command | Type of register mark | *not yet defined* |

| Attribute, Dict Entry, or Command Parameter | Type | Meaning | Registered Names |
|---|---|---|---|
| ScoringSide | dict entry in dictionary for Cover Applications | Specifies the side from which the scoring tools works; the dict entry is an array of names | /FromInside<br>/FromOutside |
| SewingPattern | dict entry in operation specific dictionary for ThreadSewing and SideSewing operation | Sewing pattern | /Normal<br>/Staggered<br>/CombinedStaggered |
| SourceType | dict entry in component definition dictionary | Specifies the source type of a product component | /Sheet<br>/Block<br>/PartialProduct<br>/ExternalProduct |
| StapleShape | dict entry in operation specific dictionary for SaddleStitching and Stitching operation | Shape of staples | /Crown<br>/Overlap<br>/Butted<br>/ClinchOut<br>/Eyelet |
| ThreadMaterial | dict entry in operation specific dictionary for SideSewing operation | Thread material | /Cotton<br>/Nylon<br>/Polyester |
| *to-direction* | parameter of one step in folding operation | Direction in which it is folded | /Up<br>/Down |
| Type | dict entry in data dictionary for color or density measuring field | type of measuring field | /CIELAB<br>/Density |
| *working-direction* | parameter of one step in folding operation | Direction from which the tool is working | /Top<br>/Bottom |

**Table D-2: Attributes, dict entries, and command parameters using registered names**

# Appendix E. List of Illustrations



# Appendix F. List of Tables

# Appendix G. List of Examples

# Appendix H. Index of Key Words and Comments

## A



## B



## C

## D



## E



## F



## G

## H



## I



## L



## M



## N



## O



## P

## R



## S



## T

## U



## W