LRS OEM Transforms
Installation and User’s Guide

Subject to License/Trade Secret

Trademarks

Copyright

Levi, Ray & Shoup, Inc.
2401 West Monroe Street
Springfield, IL 62704
Phone: 217-793-3800
Fax: 217-787-4014
http://www.lrs.com

Document Number: S666-0100-1 Revision Date: July 28, 2005

Table of Contents

Summary of Enhancements

The following table contains the fix numbers assigned to major enhancements and/or fixes.

LRS OEM Transforms V1 R1.0 Fixes	

Section 1
Overview

Introduction

The LRS OEM conversions (CONV-1) are a series of document transformations that are executed via the standard VPSX FILTER process (see the VPSX Installation and User Guide for the description of the VPSX FILTER Process) or an AnyQueue backend (see the AnyQueue Installation and User's Guide for a description of the AnyQueue Backend).

VPSX can intelligently select a process to be invoked at document print time based on the original content of the document when the dataset is captured. This process allows the execution of any SHELL or program and provides the platform the execution of the LRS OEM transforms.

The transforms can convert the input data into a completely different output data type that is subsequently processed by VPSX. This often involves the transformation from data streams that are text based (SAP OTF/GOF) into page based data streams such as POST-SCRIPT.

AnyQueue allows the execution of any SHELL or program and provides the platform for execution of the LRS OEM Transforms. After a job is sent to a route, the Backend associated with that route will be executed.

The supplied LRS OEM conversions consist of several key components:

- Presentation Area
- Mixed Format Filter

Presentation Area (PA)

The Presentation area is the general name for the intermediate or common format that documents undergoing conversion occupy.

Mixed Format Filter

The Mixed Format Filters are a series of input control files that are used to create the Presentation Area using the document format undergoing transformation and output control files that are used to create the new document format from the Presentation Area.

The MFF control files are encoded in XML and are specific to the input and output document types.

Each MFF filter is specialized in processing of a certain format, e.g., PDF, AFP, or a raster image. These filters can read and write documents of one of these formats. The input, as well as the output, processing functions make use of the same common layer - the presentation area (PA). The input processing functions of a MFF filter read the source format into the PA and the output processing functions write the target format out of the PA. Thus, the LRS OEM products are able to combine the input processing functions of any MFF filter with the output processing functions of any other MFF filter.

The information in the presentation area (PA) completely describes one page of a document. All the objects of one page (text in Unicode, raster images, lines, rectangles etc.) and, user defined codepages and special characters, provided by mixed object formats like AFP and PDF, are saved in the PA in a unique format, independent of their input and output.

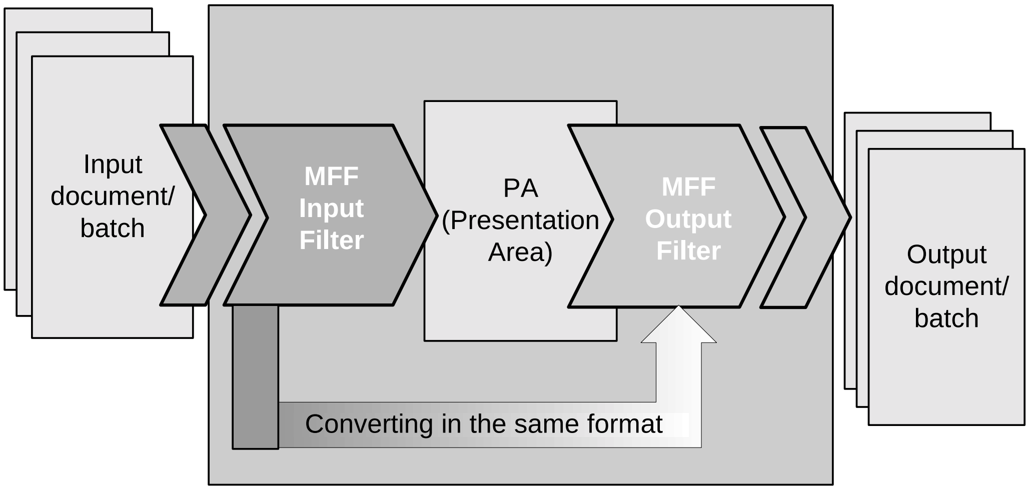


Figure 1-1: Input and Output Processing via MFF Filter and Presentation Area

The input filters enable the reading of a single page of a document or the document batch in the PA. During this process, the format specific objects are converted to generic objects of the PA. The various object types (text, raster images, lines, rectangles, etc.) remain unchanged.

The output filters write the various objects out of the PA page by page in the requested output format. Therefore, at the end, the complete document and/or document batch will be present in the target format. Depending on the output format used, the respective object type will be transferred into the related target format.

MFF Filter Overview

The LRS OEM Transforms enable the following conversion types:

LRS Product	**Platform**	**LRS Key Name**
VPSX/AFP to PCL	Unix	KEYAFP2PCL
VPSX/AFP to PDF	Unix	KEYAFP2PDF
VPSX/AFP to PS	Unix	KEYAFP2PS
VPSX/PCL to AFP	Unix	KEYPCL2AFP
VPSX/PCL to PDF	Unix	KEYPCL2PDF
VPSX/OTF to AFP	Unix	KEYOTF2AFP
VPSX/OTF to PCL	Unix	KEYOTF2PCL
VPSX/OTF to PDF	Unix	KEYOTF2PDF
VPSX/OTF to PS	Unix	KEYOTF2PS
AnyQueue/AFP to PCL	Unix/Windows	KEYAFP2PCL
AnyQueue/AFP to PDF	Unix/Windows	KEYAFP2PDF
AnyQueue/AFP to PS	Unix/Windows	KEYAFP2PS
AnyQueue/PCL to AFP	Unix/Windows	KEYPCL2AFP
AnyQueue/PCL to PDF	Unix/Windows	KEYPCL2PDF
AnyQueue/OTF to AFP	Unix/Windows	KEYOTF2AFP
AnyQueue/OTF to PCL	Unix/Windows	KEYOTF2PCL
AnyQueue/OTF to PDF	Unix/Windows	KEYOTF2PDF
AnyQueue/OTF to PS	Unix/Windows	KEYOTF2PS

VPSX

These conversions operate in the VPSX FILTER. The FILTER process is described in the VPSX Installation and User Guide.

AnyQueue

For AnyQueue these conversions operate as backend process. The backend process is described in the AnyQueue Installation and User’s Guide.

The LRS OEM transform requires specific input and output MFF FILTERS to be available in the execution directory or path to enable the transformation to occur.

Each transformation is associated with a specific LRS supplied key. The key will determine if the transformation can proceed and is specific for each and every valid combination of INPUT and OUTPUT FILTER.

The following filters and their functions are summarized. If not otherwise specified, the filters are available as both OUTPUT and INPUT filters.

MFFAFP: AFP and MO:DCA

The MFF filter MFFAFP supports the different IBM formats AFP (Advanced Function Printing) and MO:DCA (Mixed Object:Document Content Architecture). Because of the market significance of AFP in the large customer area, strong emphasis on conversion fidelity and depth. The filter supports the following features:

- Inline and external resources as well as ACIF compatible resource libraries.
- Raster fonts and related code pages.
- Overlays, page segments and medium maps.
- Pantone and true colors (e.g., JPEG images).
- TLE and NOP processing for archives and printing systems (e.g., POSI).

MFFGOF: SAPGOF

The GOF format used by SAP (GOF = Generic Output Format) includes the OTF format (Output Text Format) and the SAP list format. Since, in this case, only conversions of GOF documents into other formats are relevant, the MFF filter MFFGOF is only designed as input filter. The filter handles data created by SAP SMART FORMS, SAPSCRIPT and ABAP List Format (ALF) and is able to handle SAP BARCODES without having printer specific SIMM cards as is normally recommended by SAP. The filter supports the following features:

- Intelligent font mapping.
- Raster images.

MFFPCL: HP PCL

The MFF filter MFFPCL supports the HP format PCL (Printer Control Language). The filter supports the following features:

- Intelligent font mapping.
- Macro and download font.

MFFPDF: Adobe PDF

The MFF filter MFFPDF supports the largely accepted Adobe format PDF (Portable Document Format). Because of the market penetration of PDF in the Web community, there is a strong emphasis on conversion fidelity and depth. The filter supports the following features:

- Optimized compression.
- Web format (optimized).
- Password protection with encryption.
- Font mapping versus font inclusion for size optimizing.
- Harmonization of AFP functions like logos as fonts.

MFFPOS: PostScript

The MFF filter MFFPOS supports the PostScript format. It is designed as output filter. The PostScript format describes the appearance of text, graphics, and images on printed or displayed pages. The filter supports the following features:

- Optimized compression.
- Web format (optimized).
- Password protection with encryption.
- Font mapping versus font inclusion for size optimizing.

Supported Operating Systems

The LRS OEM transforms are available on the same platforms that VPSX currently supports. Namely,

- GNU/Linux (Kernel 2.4 and glibc 2.2 and 2.3 (e.g., SLES 7 and 8, SuSE Pro 8.2 and 9.0) (VPSX and AnyQueue).
- AIX (4.2.1, 4.3.3, and 5.2L) (VPSX and AnyQueue).
- Solaris (8 to 9) (VPSX and AnyQueue).
- HP-UX (11 on PARISC) (VPSX and AnyQueue).
- Windows 2000, XP, 2003 (AnyQueue only).

LRS OEM Conversion Program Modules

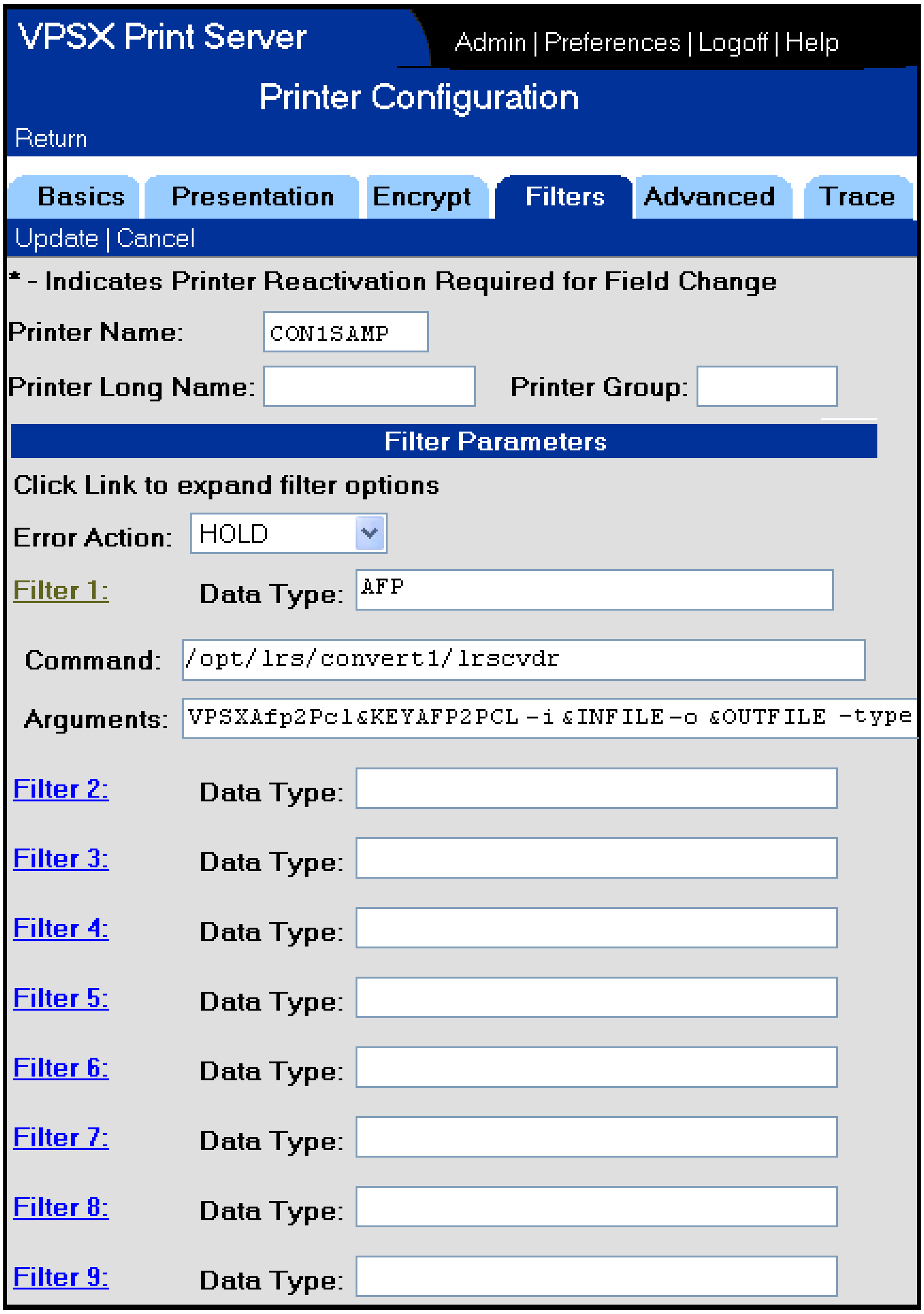
LRSCVDR

VPSX

LRSCVDR is the command that drives the LRS OEM transforms in the VPSX Filter process. The arguments passed to LRSCVDR provide for checking that a valid license for a particular pair of INPUT and OUTPUT FILTERS, effectively a specific transformation type, is valid and for the control of the transformation process. VPSX has a number of built-in system variables that can be passed as arguments to the conversion process. These are described in Appendix A of the VPSX Installation and Users guide.

The first two arguments identify the particular transform type and the associated LRS OEM transform key specified in the VPSX Administration ‘Product Keys’ screen. The arguments MUST be supplied in the order specified in the example in Figure 1-2 on page 1.8.

The LRS OEM transforms always requires input data. The &INFILE variable contains the name of a dataset that contains the data captured by VPSX. VPSX normally will print the result of the transformation process and supplies the &OUTFILE variable to the transformation. This variable contains the name of a dataset that VPSX will read after the transformation process completes, that contains the transformed input data. VPSX will normally attempt to print this file. If the transformation process fails then VPSX does not attempt to print the file described by &OUTFILE.



* - Indicates Printer Reactivation Required for Field Change

Printer Name: CON1SAMP

Printer Long Name:

Printer Group:

Error Action: HOLD

Filter 1: Data Type: AFP

Command: /opt/lrs/convert1/lrscvdr

Arguments: VPSXAfp2Pcl&KEYAFP2PCL -i &INFILE -o &OUTFILE -type

Filter 2: Data Type:

Filter 3: Data Type:

Filter 4: Data Type:

Filter 5: Data Type:

Filter 6: Data Type:

Filter 7: Data Type:

Filter 8: Data Type:

Filter 9: Data Type:

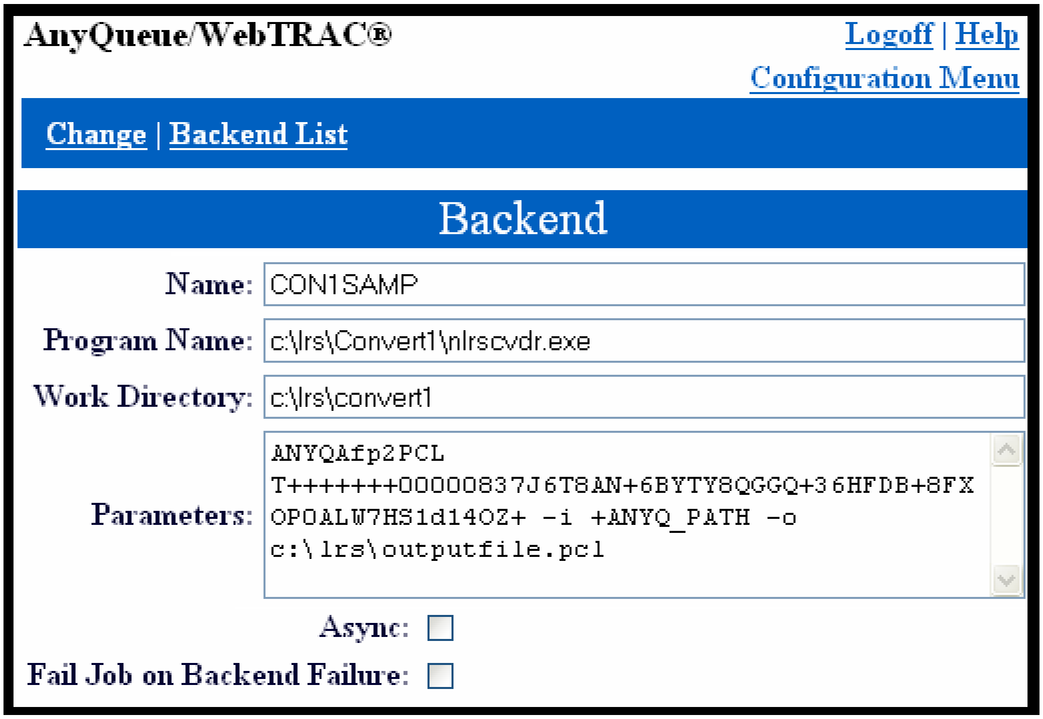
Figure 1-2: VPSX Filters

The figure above shows the VPSX FILTER definition and some of the parameters that can be passed to the LRS OEM Transform.

AnyQueue

NLRSCVDR is the command that drives the LRS OEM Transforms in an AnyQueue backend process. The arguments passed to NLRSCVDR provide checking that a valid license for a particular pair of INPUT and OUTPUT FILTERS, effectively a specific transformation type, is valid and provides the control of the transformation process. AnyQueue has a number of built-in system variables that can be passed as arguments to the conversion process. These are described in the AnyQueue Installation and User's Guide. The first two arguments identify the particular transform type and the associated LRS OEM Transform key. The arguments MUST be supplied in the order specified in the example in Figure 1-3.

The LRS OEM Transforms always require input data. The &INFILE variable contains the name of a dataset that contains the data captured by AnyQueue.



Name: CON1SAMP

Program Name: c:\lrs\Convert1\nlrscvdr.exe

Work Directory: c:\lrs\convert1

Parameters:

```
ANYQAfp2PCL
T+++++++00000837J6T8AN+6BYTY8QGGQ+36HFDB+8FX
OPOALW7HS1d14OZ+ -i +ANYQ_PATH -o
c:\lrs\outputfile.pcl
```

Async: □

Fail Job on Backend Failure: □

Figure 1-3: AnyQueue Backend Parameters

Section 2
Installation

Installation onto the Target System

The installation process uses the standard LRS product installation process as described in the VPSX Installation and Users Guide and AnyQueue Installation and User's Guide.

Unix Platform

The LRSINST program can be used to install all the necessary components required to execute the transforms. The installation 'Product-Set' CON1 will install all transforms described in this manual (example: ./lrsinst con1). The transforms are activated by the use of a specific transformation type, a transformation key, and a transformation license file. The transformation type refers to the type of transformation that will be performed (example: VPSXAFP2PCL, or ANYQAFP2PCL). This value must be specified as the first positional argument to the LRSCVDR program. The second position parameter specifies the 60 character product key for the requested transformation. (When executing transformations under VPSX a symbolic variable can be used to provide this value that will be dynamically replaced with the key value specified in the VPSX system configuration file (example: &keyafp2pcl)). Finally, the transformation license file must be copied to the product installation directory (default /opt/lrs/convert1). The license file is provided in the root directory of the product distribution material and the file name will match the transformation type with the extension '.lic' appended (e.g., vpsxafp2pcl.lic).

Windows Platform

LRSCONV1.EXE is a self-extracting Installshield program that is used to install all the necessary components required to execute the transforms.

VPSX

The installation process for VPSX will create a sample printer definition called **con1samp** that contains an example of the filter definitions required to execute these conversion routines.

As part of the installation process it may be necessary to customize the configuration files for the specific input and output types being used. Sample configuration files for all input and output types are provided in the product installation directory with a suffix of '.pr~' (example: mffafp.pr~). To use these samples it is first necessary to rename distribution files to '.pro' (e.g., mffafp.pro). The configuration files contain profile statements encoded in XML that control the processing of the associated input/output data type. For complete details of the configuration options available please refer to "Configuration" on page 3.1.

Command Line Parameters

The LRS OEM Transforms are executed using the LRS conversion driver routine (LRSCVDR on Unix, and NLRSCVDR on Windows). This routine validates the transformation type and key values that are specified as the first two positional parameters and then executes the appropriate conversion routine. All arguments following the transformation type and key arguments are passed directly to the requested transformation routine.

Example:

Unix

```
lrscvdr VPSXAFP2PCL &keyafp2pcl -i &infile -o &outfile
lrscvdr ANYQAFP2PCL <60byte key> -i +ANYQ_PATH -o /opt/lrs/outputfile.pcl
```

Windows

```
nlrscvdr ANYQAFP2PCL <60byte key> -i +ANYQ_PATH -o c:\lrs\outputfile.pcl
```

Additional transformation specific arguments can be added to the end of the above commands.

lrscvdr Command Line Parameters

lrscvdr is called with the following arguments or parameters:

convtype convkey -i *inputfile1* [*inputfile2...*] -o *outputfile*

Parameters

According to their position in the command line, parameters can be classified:

- Positional parameters
- Global parameters
- Input parameters, specified after **-i**
- Output parameters, specified after **-o**

Notes:

1. Some parameters can be specified for the input as well as for the output process. But input parameters are processed first.
2. The parameters can be abbreviated, but they must be unique. The parameters are listed alphabetically.

Positional Parameters

There are two positional parameters that must be specified as the first two arguments in the execution of LRSCVDR. The **CONVTYPE** keyword must be the first keyword and the **CONVKEY** parameter will be the second keyword. The CONVTYPE keyword will refer to a license file that must be present in the transformation installation directory (this was explained previously in the installation instructions).

These are the only valid combinations of CONVTYPE and CONVKEY and the associated license file:

CONVTYPE	**CONVKEY**	**License File name**
VPSXAFP2PDF	&KEYAFP2PDF	vpsxafp2pdf.lic
VPSXAFP2PS	&KEYAFP2PS	vpsxafp2ps.lic
VPSXPCL2AFP	&KEYPCL2AFP	vpsxpcl2afp.lic
VPSXAFP2PCL	&KEYAFP2PCL	vpsxafp2pcl.lic
VPSXOTF2AFP	&KEYOTF2AFP	vpsxotf2afp.lic
VPSXOTF2PCL	&KEYOTF2PCL	vpsxotf2pcl.lic
VPSXOTF2PS	&KEYOTF2PS	vpsxotf2ps.lic
VPSXOTF2PDF	&KEYOTF2PDF	vpsxotf2pdf.lic
ANYQAFP2PDF	&KEYAFP2PDF	anyqafp2pdf.lic
ANYQAFP2PS	&KEYAFP2PS	anyqafp2ps.lic
ANYQPCL2AFP	&KEYPCL2AFP	anyqpcl2afp.lic
ANYQAFP2PCL	&KEYAFP2PCL	anyqafp2pcl.lic
ANYQOTF2AFP	&KEYOTF2AFP	anyqotf2afp.lic
ANYQOTF2PCL	&KEYOTF2PCL	anyqotf2pcl.lic
ANYQOTF2PS	&KEYOTF2PS	anyqotf2ps.lic
ANYQOTF2PDF	&KEYOTF2PDF	anyqotf2pdf.lic

Global Parameters

[-bigjob]
-infile *input.ext*
[-licfile *file*]
[-logdate +/-]
[-logdir *path*]
[-logfile *file*]
[-logtime +/-]
[-modulepath]
-o *output.ext*
[-profiledir *profile directory*]
[-quiet +/-]
[-stdin]
[-stdout]
[-stdoutdirect]
[-verbose +/-]

Input Parameters

[-bwc *matrix size*]
[-bwd *matrix size*]
[-bwt *percent*]
[-cdr.hideabmps]
[-cdr.aprocmode *name*]
[-charsetpath *path*]
[-converttoimage]
[-copies *number*]
[-dx *length*]
[-dy *length*]
[-embedfilespath *path*]
[-formdef *form definition*]
[-formdefpath *path*]
[-gentoc]
[-genthumbnails]
[-jobname *name*]
[-macrodir *path*]
[-page *m..n*]
[-pwi *name*]
[-remove *i*|*l*|*r*|*t*|*p*|*a*]
[-reslibin *name*]
[-reverseduplex]
[-rotate *n*]
[-systemfontpath *path*]
[-tray *trayname*]
[-type *type.subtype*]

Output Parameters

[-addpagesforduplex]
[-afp.generateoverlays]
[-afp.writenods]
[-autoformdef]
[-charsetpath *path*]
[-cx *pixels*]
[-cxmax *pixels*]
[-cy *pixels*]
[-cymax *pixels*]
[-converttoimage]
[-copies *number*]
[-dx *length*]
[-dy *length*]
[-embedfonts]
[-formdef *form definition*]
[-genthumbnails]
[-iff.extractimages]
[-jobname *name*]
[-linearized +/-]
[-nobinarycopy]
[-pagesize *papersize*]
[-pdf.nochange]
[-pdf.textnotes.nochange]
[-pdf.copy.notallowed]
[-pdf.print.notallowed]
[-pdf.writeuncompressedstreams]
[-printtofile]
[-prt.scalemode *]
[-pwo *name*]
[-pwu *name*]
[-remove *i*|*l*|*r*|*t*|*p*|*a*]
[-reducex *length*]
[-reducey *length*]
[-reslibin *name*]
[-rotate *n*]
[-rx *resolution*]
[-ry *resolution*]
[-scaletogray]
[-sd **]
[-selfcontained]
[-splitmask *mask*]
[-stamp *file*]
[-suppress]
[-tray *trayname*]
[-type *type.subtype*]
*   original|page
*** simplex|duplex*

Parameter Description

The parameters with their description are listed alphabetically.

Parameters	Description
addpagesforduplex	Add pages at the end of the input document, if the page number is odd.
afp.generateoverlays	Generate AFP overlays, additionally **-selfcontained** must be specified.
afp.writenods	No DS byte is written into AFP output file.
autoformdef	Activate automatic formdef generation for AFP files. Formdefs can be generated with the following information: tray, number of copies, and simplex/duplex.
bigjob	Files are processed sequentially. When program starts, the existence of the input files is not checked.
bwc	Clustered order dithering with specification of matrix size (2 - 8); default: 3.
bwd	Dispersed order dithering with specification of matrix size (2 - 8); default: 4.
bwt	Monochrome conversion with specification of threshold value (0 - 100) in percent; default: 50.
cdr.hideabmps	Hide of attachments icons of CDR files.
cdr.aprocmode	Processing mode for attachments of CDR files; options: **nop, inplace, append, only**; default: **inplace**.
charsetpath	Directory path to location where fonts are stored.
converttoimage	Convert pages to bitmaps. Vector fonts cannot always be rendered.
copies	Copies per page. See **autoformdef**.
cx	Size of output bitmap in x direction; unit: pixel.*
cxmax	Maximum size of output bitmap in x direction; unit: pixel.*
cy	Size of output bitmap in y direction; unit: pixel.*
cymax	Maximum size of output bitmap in y direction; unit: pixel.*
dx	Shift page contents in x direction.
dy	Shift page contents in y direction.
embedfonts	Embedding of fonts in output file according to the profile specification.
formdef	Formdef(inition). Input file: Formdef used, but it overwrites specifications in element **<defaultformdef .../>** in profile **mffafp.pro**. Output file: Formdef written in AFP file. See **selfcontained**.
formdefpath	Directory path to location where formdefs are stored.
gentoc	Generate table of contents in output file (PDF). Currently it is always generated with TLEs (Tag Logical Element) of an AFP file.
genthumbnails	Generate thumbnails in output file (PDF).

Parameters	Description
iff.extractimages	Extract bitmaps to a multi-page file. Output type must be a multi-page image format, e.g., iff.tif.
infile	Specify an input file group and parameters. All parameters in the command line that are specified after **-infile** up to the next **-infile** or **-outfile** are applied to all files specified in this range.
jobname	Name of print job (PRT).
licfile	Name of license file; Default: **mcopy.lic** or **cpmill.lic**.
linearized	Generate a 'linearized', i.e., web-enabled file. If Acrobat Reader browser plug-in is used, a single page of a file can be requested from a Web server and displayed (only supported by MFFPDF filter).
logdate	Write date to log file.
logdir	Directory where log file is stored; default: current directory.
logfile	Name of log file; default: **mcopy.log**.
logtime	Write time to log file.
modulepath	Directory path to location where the MFF filters are stored.
nobinarycopy	No binary copy. If the input files are 'converted' in the same format as the input format, identical copying (binary copying) is executed.
nobinarystamp	See **nobinarycopy**.
outfile	Specify output files and parameter. As an alternative to the file name you can specify: **-splitmask**, **-stdout**, **-type**.
page	Page interval of the input files to be processed. Possible are **-page x** for one page; **-page x..**, to process all pages from page x upward; **-page ..y**, to process all pages till page y; and **-page x..y**, to process the page interval from page **x** till page **y**. As an alternative to the last page you can specify **-page last**. If the number of the first page is greater than the second, the page sequence of the interval can be reversed.
pagesize	Specify page size for output file, no page scaling. The parameter value is the paper size. See Appendix B, "Page Formats" on page B.1.
pdf.nochange	Change of output file data not allowed.
pdf.textnotes.nochange	Change of comments not allowed.
pdf.copy.notallowed	Copy and paste of contents not allowed.
pdf.print.notallowed	Print of contents not allowed.
pdf.writeuncompressedstreams	Write PDF data stream uncompressed.
printtofile	Write data stream in file, only for MFFPRT filter (OS/2, Windows).
profiledir	Directory where profile files (**mffxxx.pro**) are searched. If the parameter is omitted, profile files are searched in the directory where filter modules (DLLs) are stored.
prt.scalemode	Specify scaling; options: **original, page**; default: **original**.

Parameters	Description
pwi	Password for input files.
pwo	Owner password for output files.
pwu	User password for output files.
quiet	No output displayed in command line.
reducex	Reduce page width to specified value.
reducey	Reduce page height to specified value.
remove	Remove all elements of a certain type from page. Element type are: **a** = arc, **i** = images, **l** = lines, **r** = rec(tangle)s, **p** = path, **t** = text.
reslibin	Additional resource library for AFP resources. See "MFFAFP - Resources" on page 3.8.
reverseduplex	Invert duplex information, i.e., back get front and vice versa.
rotate	Rotate to main page orientation and automatic rotation of wrong oriented pages. Options are a **multiple of 90 degree angles**, **auto**, **landscape**, and **portrait**.
rx	Specify resolution of output file in x direction in dpi (dots per inch). Supported for AFP and bitmap formats.*
ry	Specify resolution of output file in y direction in dpi (dots per inch). Supported for AFP and bitmap formats.*
scaletogray	Convert output page into 16 grayscales, not black/white, only for pages in bitmap format. Font display is improved.
sd	Simplex-/duplex information; default: simplex.
selfcontained	Write self-contained AFP files. Required (inline) resources are included into output file.
splitmask	Specify a mask for output file name. C-printf mask is used to form the output file name (0-based; page counter starts with 0).
stamp	Add stamp or watermark. Multi-page stamp file is supported. Last page of a stamp is used for following pages.
stdin	Read input data stream from **stdin**.**
stdout	Write output data stream to **stdout**.**
stdoutdirect	Write output data stream without buffering to **stdout**. Option cannot be specified for all filters.**
suppress	Suppress blank pages in output file.
systemfontpath	Directory path to location where system fonts are stored.
tray	Specify tray name that contains paper to be used. For AFP and PCL, the name assignment to tray number is specified in the related profile.

Parameters	Description
type	Enforced type of input or output file; format is type.subtype. Type specification of input file is option because type is automatically detected.
	Available input filters (= type or subtype) are:
	• AFP, available subtypes:
	• GOF: SAP format OTF
	• MOD: MO:DCA
	• PCL
	• PDF
	Available output filters:
	• AFP
	• PDF
verbose	Detailed information about converted files.
Notes: *	Parameters can be used to generate thumbnails and specify their size and resolution.
**	See examples on page 2.10.

Example

Perform the conversion of an input file. The input file is AFP and the output format file is PCL. The KEY is the KEYAFP2PCL defined in the VPSSTART member. The input file can be found in an input file with a symbolic name of &INFILE and the output should be written to the OUTPUT file defined by the symbolic variable &OUTFILE

Conversion log messages are placed in the temporary directory defined by the symbolic variable &TMPDIR and the LOG files have the same name as the printer name.

```
Lrscvdr VPSXAfp2Pcl &KEYAFP2PCL -i &INFILE -o &OUTFILE -type pcl -logdir
&TMPDIR -logfile &PRINTER -quiet
```

Section 3
Configuration

Customizing the Profiles

The functioning of the LRS OEM transforms is based on the content of the MFF FILTER profiles that are encoded in XML.

MFF Filter Settings

The MFF input and output filter are part of the MFF architecture and are used to read and write files in different formats. They can be customized be means of profiles in XML format to the environment in which the application is running.

In the profiles it is possible to specify the locations where certain resources are located, what codepages must be used and how fonts are assigned. Although the profiles of different MFF filters are similar in some aspects, they are described separately to allow a better overview about all relevant sections.

None of the MFF filters requires a profile. Profiles are only used when the conversion results shall be improved.

If there is nothing else specified in the profile during reading of a page with an MFF filter, the font name is written into the Presentation Area, as it is specified in the input file. During writing a page, a certain font in the Presentation Area can be assigned to any font. If, during reading of an AFP file, the font C0OCB11 is referenced, this font can be replaced by the font OCR-B for the output in PDF.

In most cases, the automatic font matching of the writing MFF filter is sufficient to get practical conversion results. The writing filter searches a suitable font corresponding to the font attributes Serif/Sans Serif, Proportional/Monospaced, and Italic/not Italic.

MFFAFP

MFFAFP - Global Settings

In the following, global settings are described, which overwrite the predefined filter settings.

Example: Global settings in the MFFAFP profile

```
<globals>
  <defaultchar           unc="003F"    />
  <multidocument         value="true"  />
  <generateoverlays      value="false" />
  <autoformdefgeneration  value="false" />
  <treatrectstransparent  value="false" />
  <rasterunevenpointsizes value="false" />
  <rastersettype          value="screen"/>
  <overlayasreference     value="false" />
  <pagesegmentasreference value="false" />
  <mappagesegments        value="false" />
  <optimizepa             value="always"/>
  <aliasmap  devicename="SD150300"
             map="SD150000" condition="ifnotfoundinencodings"/>
             <!-- U+02C6: Circumflex Accent -->
  <aliasmap devicename="SD190300"
             map="SD190000" condition="ifnotfoundinencodings"/>
             <!-- U+007E: Tilde -->
  <aliasmap  devicename="SD310000"
             map="SM150000" condition="ifnotfoundinencodings"/>
             <!-- U+00AF: Macron -->
  <aliasmap  devicename="GF010002"
             map="GF010000" condition="ifnotfoundinencodings"/>
             <!-- U+03D5: Phi Small (open form) -->
  <aliasmap  devicename="SS680000"
             map="SP100000" condition="ifnotfoundinencodings"/>
             <!-- U+2013: En Dash -->
  <aliasmap  devicename="ND051001" map="ND051000"
             condition="ifnotfoundinencodings" />
             <!-- U+2075: Five Superscript -->
```

<defaultchar.../>

Element specifies the character, which is written, if the appropriate character is not available, i.e., if it was not found in a codepage. The value of the attribute **`unc`** is a hexadecimal number that represents the character.

<multidocument.../>

Element specifies whether only one or several documents, which are generated in one file, should be read, i.e., it applies to files, in which many AFP overlays or page segments are located. The values of the attribute **`value`** are **`true`** or **`false`**.

<generateoverlays.../>

While writing is executed, element controls whether references to "external pages" in form of an AFP overlay will be generated in the output file (inline resource). These "external pages" can be AFP overlays that were not resolved during reading (see element <**`overlayasreference.../`**), or overlays ("stamp") with other formats that were contributed during the conversion. The values of the attribute **`value`** are **`true`** or **`false`**.

<autoformdefgeneration.../>

Element controls the automatic generation of AFP copy groups (= Medium Maps) with data like tray, number of copies, and simplex/duplex. This information can come from the reading filter (e.g., from PCL) during conversion or can be set by the user.

For any new arising combination of the information mentioned above a new copy group will be created. All automatically generated copy groups will be numbered and stored in the formdef '**`F1CPAUTO`**'. As in the case of automatic overlay generation, inline resources will be generated. The values of the attribute **`value`** are **`true`** or **`false`**.

<treatrectautoformdefgeneration.../>

Element controls AFP generation, especially when converting PDF files. For tables in PDF files often the following elements are generated: At first (mostly black) lines and subsequently table cells with non-transparent rectangular areas (sometimes with filling patterns) that overlap the lines partially. This normally causes the rectangular areas to be automatically combined with the lines to a bitmap, because of AFP.s character pixel can only be made black and never white (the area overlapping the line must not be black, but must have the filling pattern of the cell).

Since the output file will become larger because of the conversion into a bitmap and the result will not have a better quality in any case, it is possible to treat the rectangular areas always as transparent areas to prevent any conversion. The values of the attribute **`value`** are **`true`** or **`false`**.

<treatpathstransparent.../>

Element controls whether paths are treated as transparent paths. The values of the attribute **`value`** are **`true`** or **`false`**. The default value of the attribute **`value`** is **`false`**. For more information, see the description of element **`<treatrectstransparent.../>`**.

<rasterunevenpointsizes.../>

Note: In documents with formats that support vector fonts, e.g., TrueType or Type 1, (PDF or PCL) the following problem comes up when converting to a format that does not support vector font like AFP: Non-integer font sizes (e.g., 10.25) cannot be reproduced exactly.

To eliminate differences of visual integrity, element controls converting text with non-integer font sizes into bitmaps. Note: Visual integrity improves at the expense of the file size. The values of the attribute **`value`** are **`true`** or **`false`**.

<rastersettype.../>

If areas are written with filling patterns, different types of filling patterns are available. Element is used to provide a specified area with a defined pattern type. The values of the attribute **`value`** are **`screen`**, **`screenmodified`**, **`afp`**, **`afpmodified`**, **`roughw0`**, **`roughw15`**, **`roughw45`**, **`roughw75`**, **`fine0`**, **`fine15`**, **`fine45`** and **`fine75`**.

<overlayasreference.../> / **<pagesegmentasreference.../>**

Note: For the reading process of an AFP file, the user can be faced with the following question: Should referenced overlays be maintained as they are or should the information from the overlays be resolved into the page? There are several options, to control this behavior during reading as well as during writing into AFP.

While reading is executed, element controls whether overlays/page segments that are referenced on an AFP page shall be resolved into the page or the references to the overlays/page segments shall be treated as they are.

If the references are treated as they are, there is a possibility to write the content of the overlay only once (although several references do exist) when writing in appropriate output formats. Currently, AFP filter and PDF filter are able to generate information from overlays only once.

The values of the attribute **`value`** are **`true`** or **`false`**.

<mappagesements.../>

Element controls, whether Map Page Segment structured field (MPS) is processed. The values of the attribute **`value`** are **`true`** or **`false`**. If **`true`** is defined, page segments are loaded into the printer and stored until the print job is completed (hard object). If **`false`** is defined, page segment are sent to the whenever and as often as requested but not stored (soft object).

<optimizepa.../>

Element controls the generation of the page before it is written to AFP. The values of the attribute **`value`** are **`ifdifferenttypes`**, **`always`** or **`never`**, **`ifdifferenttypes`** must be specified, if the input filter is not the AFP filter. Then, the page is generated for AFP output.

<aliasmap.../>

Element controls whether a character described by a character identifier is replaced by another character. If a character is not found in a specified code page during writing, a defined replacement character is used, see “MFFAFP - Code Pages” on page 3.7.

The values of the attribute **`condition`** are **`ifnotfoundinencodings`** and **`always`**. The values specified for the attributes **`devicename`** and **`map`** is an IBM graphic character identifier.

MFFAFP - User Defined Encoding

Normally in the text of AFP files, a byte represents a particular character. However to be able to display more than 256 characters, each text section is assigned to a certain code page. This code page assigns each byte in the text (code point) to a character. There is a set of default character names, but of course, they do not cover all characters. Not each IBM specified name is assigned to a Unicode character and inversely not all Unicode characters have a representation with the default character names. In addition, many AFP users create their own characters. Because the MFF architecture always stores text in Unicode it is possible to fix the relationship between a GCGID and a UNICODE value in the MFFAP profile.

Example: User defined encoding in the MFFAFP profile

```
<encodinglist>

  <encoding name="T1ANSI">

    <entry devname="TRANS032" unc="0020"/>  <!-- space                -->
    <entry devname="TRANS033" unc="270F"/>  <!-- pencil               -->
    <entry devname="TRANS034" unc="2702"/>  <!-- black scissors       -->
    <entry devname="TRANS035" unc="2701"/>  <!-- upper blade scissors -->
    <entry devname="TRANS040" unc="260E"/>  <!-- black telephone      -->
    <!-- [...]                         -->
  </encoding>

  <encoding name = "elixirsymbol">
    <entry devname="XC065410"  unc="F041"/>  <!.s of a saving bank-->
  </encoding>

<encodinglist>
```

MFFAFP - Fonts

Normally, AFP fonts are available as raster fonts. You must specified which font file should be used for which font size.

Example: Font definition in the MFFAFP profile

```
<fontlist>
   <font family="Arial" serifstyle="SANSSERIF" spacing="PROPORTIONAL" >

     <face   weight="MEDIUM"
             width="NORMAL"
             style="UPRIGHT"
             baselineincrement="1150">

       <raster devname="C0A48410" size= "1" />
       <raster devname="C0A48420" size= "2" />
       <raster devname="C0A48430" size= "3" />
       <raster devname="C0A48440" size= "4" />
       <raster devname="C0A48450" size= "5" />
     </face>

     <face   weight="BOLD"
             width="NORMAL"
             style="UPRIGHT"
             baselineincrement="1150">

       <raster devname="C0A58510" size= "1" />
       <raster devname="C0A58520" size= "2" />
       <raster devname="C0A58530" size= "3" />
       <raster devname="C0A58540" size= "4" />
       <raster devname="C0A58550" size= "5" />
     </face>
   </font>
</fontlist>
```

MFFAFP - Code Pages

Example: Code page definition in the MFFAFP profile

```
<codepagelist>
   <codepage name="T1V10273" iana="IBM273" default="yes"/>
   <codepage name="T1V10500" iana="IBM500" />
</codepagelist>
```

<codepagelist>

The subelement **`<codepage.../>`** of the element **`<codepagelist...>`** defines several code pages. One of them can be assigned by the attribute `default` with the value `yes` as the default code page. When writing to AFP, the code page will be switched automatically as long as characters are not found in the currently used code page.

MFFAFP - Resources

AFP resources can either be bundled in one file (as resource library) or exist as single resources.

Example: Resource definition in the MFFAFP profile

```
<resourcelist>
  <library filename="resources1" />
  <library tle="reslib"          />
  <files path="\\fileserv\drive\resoure\reslibs"    type="resourcelib"
                                                    extension="" />
  <files path="\\fileserv\drive\resourc\codedfonts" type="codedfont"
                                                    extension="300" />
  <files path="\\fileserv\drive\resourc\charsets"   type="charset"
                                                    extension="" />
  <files path=\\fileserv\drive\resourc              extension="" />
</resourcelist>
```

<resourcelist>

Within the element all resource definitions are specified. To find resource libraries ("Reslibs") only file name and path must be known. The file name can be fully qualified, **`<library filename="/home/user/afpres1.lib"/>`** or without path which can be specified for several Reslibs together, **<files path="/home/user" type="resourcelib" extension="lib"/>**.

It is also possible to pick up a name of a Reslib from a TLE during reading an AFP file, e.g., **`<library tle="reslib"/>`**.

To specify the location, which should be scanned for AFP resources, it is sufficient to specify the directory, in which the resources are **`<files path="/usr/local /afpres"/>`**.

Additionally by means of the attribute type, it can be specified that in the defined directory a scan process shall be executed for resources. The attribute extension restricts the search to files with a special extension. For the attribute type the values **overlay, pagesegment, codedfont, codepage, charset, formdef, resourcelib** and **pagedef** can be specified.

MFFAFP - Trays

Input trays are assigned to the input and output format via unique tray names. In AFP, tray names are specified inside a Copy Group via a number (the so-called device-ID). Because the numbers specified in different data streams are not related to each other, these numbers has to be assigned to a tray name on the input side; on the output side they again will be reconverted to a number. If in a PCL input data stream a tray is assigned to number 3, this tray number can first be assign for example to the name “PAGE1” (in the profile of the PCL module). For the output in AFP this name will be assigned again for example to the tray number 0. The release of a tray number in AFP can only be arranged, if the element **<autoformdefgeneration.../>** is specified. In this case, a Copy Group is created for the required tray number (as well as all other specifications like simplex/duplex, etc.).

Example: Tray definition in the MFFAFP profile

```
<traylist>
<!-- = inputtray    ================================================
-->
  <inputtray deviceid='1' name="UpperTray"       />
  <inputtray deviceid='2' name="ManualFeed"      />
  <inputtray deviceid='3' name="ManualEnvelope"  />
  <inputtray deviceid='4' name="LowerTray"       />
  <inputtray deviceid='5' name="LargeCapacity"   />
  <inputtray deviceid='6' name="Envelope"        />
  <inputtray deviceid='7' name="Automatic"       />
</traylist>
```

<inputtray.../>

Element specifies with the attributes **deviceid** (tray number) and **name** (tray name) the tray definition for input and output.

MFFAFP - Color Assignment

Any RGB color can be assigned to an OCA color index (AFP input).

Example: Color Index List in the MFFAFP profile

```
  <colorindexlist>
     <!--                          a r g b                         -->
     <colorindex index='0'  color="#00000000" />  <!-- DEVICE DEFAULT -->
     <colorindex index='1'  color="#000000FF" />  <!-- BLUE           -->
     <colorindex index='2'  color="#00FF0000" />  <!-- RED            -->
     <colorindex index='3'  color="#00FF00FF" />  <!-- MAGENTA        -->
     <colorindex index='4'  color="#0000F800" />  <!-- GREEN          -->
     <colorindex index='5'  color="#0000B0B0" />  <!-- CYAN           -->
     <colorindex index='6'  color="#00FFFF00" />  <!-- YELLOW         -->
     <colorindex index='7'  color="#00FFFFFF" />  <!-- WHITE          -->
     <colorindex index='8'  color="#00000000" />  <!-- BLACK          -->
       .
       .
       .
     <colorindex index='16' color="#00800000" />  <!-- BROWN          -->
  </colorindexlist>
```

<colorindex.../>

The subelement `<colorindex.../>` of `<colorindexlist>` with its attributes `index` and `color` is used to assign a RGB color to OCA color index (AFP).

Example: Highlight Color (Spot Color) List in the MFFAFP Profile

```
  <highlightcolorlist>
     <highlightcolor name='default' type='oca' number='0x0000'
                     coverage='0'  shading='0'/> <!-- default color -->
     <highlightcolor name='color 1' type='oca' number='0x0001'
                     coverage='10' shading='11'/> <!.1st hilight color -->
     <highlightcolor name='color X' type='oca' number='0x0002'/>
  </highlightcolorlist>
```

<highlightcolorlist.../>

The subelement **`<highlightcolor>`** of **`<highlightcolorlist>`** with its attributes **`name`**, **`type`**, **`number`**, **`coverage`** and **`shading`** defines a output highlight color.

If the attribute **`name`** is specified with the value **`default`**, the default color is defined. Any additional color can specified with an arbitrary name. The values of the attribute **`type`** are **`oca`** and **`highlightcolor`**. If the value **`oca`** is specified, the value of the attribute **`number`** specifies a defined set (PT1 Subset) of colors. The value of the default printer color is **`X'0000'`**.

Note: The attribute **`coverage`** and **`shading`** are not supported in combination with the value **`oca`** of the attribute **`type`**.

For information about the set of colors see profile file **MFF.PRO** and **IBM Data Stream** and **Object Architectures Presentation Text Object Content Architecture Reference.**

If the value highlightcolor is specified for the attribute type, the value of the attribute number defines a printer dependent highlight color. The value of the default printer color is **`X'0000'`**.

If the attribute **`coverage`** is specified, a percentage value (0 - 100) defines the color coverage. If the attribute **`shading`** is specified, a percentage value (0 - 100) defines the color shading. The sum of both values must be less or equal 100.

MFFAFP - Input AFP Files Settings

The following elements can be used in the element **`<input>`**.

<defaultformdef.../>

Element specifies with the value of the attribute **`name`** the formdef name, if it is not passed by the application or it is not known.

<usefontspacecharacter.../>

Element controls the use of the space character increment. The values of the attribute **`value`** are **`false`** and **`true`**. If **`false`** is specified, the space character increment parameter is used. If **`true`** is specified the font character metrics information of a character is used, see **IBM Font Object Architecture (FOCA) Reference**.

<usegridwidth.../>

Element defines whether the width value specified in the global resource identifier (GRID) is used. The values of the attribute **`value`** are **`false`** and **`true`**. If **`true`** is specified, the GRID defined width is used. If **`false`** is specified, the width value is taken form a built-in table.

<inputinterpretationmode.../>

Element defines the page size and page position. The values of the attribute **`value`** are **`page`** and **`medium`**. If `page` is specified, each page is read separately. If **`medium`** is specified, the pages are read as if they were output to the defined medium with the formdef taken into account.

<readarchive fields.../>

Element controls whether archive fields are read. The values of the attribute **`value`** are **`true`** and **`false`**. The default value is **`true`**, i.e., archive fields (TLEs) are read. If **`false`** is specified, archive fields are ignored.

<readcomments.../>

Element controls whether information in NOPs are read as comments. Note: NOP is a container for meta data. The values of the attribute **`value`** are **`true`** and **`false`**. The default value is **`true`**, i.e., data content of NOPs is read. If **`false`** is specified, content reading is ignored.

<addcommentstopage.../>

Element controls whether NOP data content is imported. The values of the attribute **`value`** are **`true`** and **`false`**. The default value is **`false`**, i.e., import of data content is ignored. If **`true`** is specified, data content is imported. Note: Element is used is such a way as to import and provide NOP data content when non-binary files are copied.

MFFAFP - Output AFP Files Settings

The following elements can be used in the element **`<output>`**.

<goca.../>

Element controls whether the output is generated with GOCA (Graphics Object Content Architecture). The values of the attribute value are false and true. The default value of the attribute value is false, i.e., GOCA generation is not performed.

If the value of the attribute value is true, preselected objects are written as GOCA objects. Objects are circles, segments of circles, lines, rectangles and text. For further definitions and settings see CPAIR.H file.

<colorspaces.../>

Element controls how color is applied to GOCA objects. The values of the attribute **`type`** (pattern) are **`FILL`** and **`STROKE`**. The values of the attribute **`value`** (color model) are **`CIELAB`, `CMYK`, `HIGHLIGHT`, `OCA`, `OCAEXT`,** and **`RGB`**. One, several or all models can be specified. If a model is selected as "best case" but not allowed to be used, an error message is issued.

<positioningmethod.../>

Element controls the positioning method for GOCA characters. The values of the attribute **`value`** are **`font`** and **`cell`**. If **`font`** is specified, the character cell is ignored and the size of the font is used to define the size of the character. If **`cell`** is specified, the size of the character cell defines the size of the character, see **IBM Graphics Object Content Architecture (GOCA) Reference**.

<writecrlfafterstructuredfield.../>

Element controls whether each structured field is followed by a CRFL. The values of the attribute **`value`** are **`false`** and **`true`**.

<avoidcrlfinsidestructuredfields.../>

Element controls the encounter of **`X'0D'`** and **`X'0A'`** resulting in the hexadecimal combination **`X'0D0A'`** (Carriage Return Line Feed) within valid AFP data, i.e. within a Structured Field. The result is that data are defective. Note: Element can be used in an ISIS software environment in combination with the file transfer or reblocking process. The values of the attribute **`value`** are **`false`** or **`true`**. The default value of the attribute **`value`** is **`false`**.

<usetrayasmediummap.../>

Element controls the use of tray names. The values of the attribute **`value`** are **`false`** and **`true`**. If **`true`** is specified, the application defined tray name is converted to a reference to a copy group. The IMM (Invoke Medium Map) structured field is generated. If **`false`** is specified, the tray is used for formdef generation.

<writeorder.../>

Element defines the writing order of **`TLE`s, `NOP`s** and **`IMM`s** in the attribute **`value`**.

<usegridwidth.../>

Element defines whether the width value specified in the global resource identifier (GRID) is used. The values of the attribute **`value`** are **`false`** and **`true`**. If **`true`** is specified, the GRID defined width is used. If **`true`** is specified, the width value is taken form a built-in table.

<maxrecordsize.../>

Element defines the maximum AFP record size. The default value for the attribute **`value`** is **`0`**, i.e., 32 KB.

<writeonlyasolutemoves.../>

Element defines whether absolute positioning commands are used exclusively or relative positioning commands are used, as well. The values of the attribute **value** are **false** for absolute and **true** for relative positioning.

<characteraccuracy.../>

Element controls that while output text is written the character spacing of the input font is compared with spacing of the output font. If the spacing is different to the specified value of the attribute value, subsequent text is repositioned. If the value **0** is specified, default values of the driver are used.

<combinexaccuracy.../>

Modifications of output font insert positioning commands into the data stream, see element **<characteraccuracy.../>**. Depending on the application element controls the suppression of negligible positioning commands. The value of the attribute **value** is the threshold value for the horizontal (x) direction (character spacing).

<combineyaccuracy.../>

Modifications of output font insert positioning commands into the data stream, see element **<characteraccuracy.../>**. Depending on the application element controls the suppression of negligible positioning commands. The value of the attribute **value** is the threshold value for the horizontal (y) direction (baseline).

<jpegquality.../>

Element controls visual integrity. The value of the attribute **value** contains a percentage number. The default value is **75**. Note: Visual integrity improves at the expense of the file size.

<monochromecompression.../>

With the attribute **value** element specifies the compression method for bi-level (two color) bitmaps. The values of the attribute are **MMR** (CCITT Modified Modified Read - IBM MMR) and **FAXG4** (CCITT T.6 Facsimile Coding - G4MMR).

<gammalist/>

Element controls the gamma value correction/tone correction. The element **<gammavalue>** defined as subelements of **<gammalist>** contains the tone correction values for rasterized areas. The attribute **from** specifies the input value and the attribute **to** the output value.

Example: Gamma value/Tone correction list in the MFFAFP profile

```
<gammalist>
          <gammavalue from=  '0' to=  '0'/>
          <gammavalue from=  '7' to= '48'/>
          <gammavalue from= '15' to= '72'/>
          <gammavalue from= '23' to= '88'/>
            .
            .
            .
          <gammavalue from='199' to='224'/>
          <gammavalue from='207' to='232'/>
          <gammavalue from='215' to='232'/>
          <gammavalue from='223' to='240'/>
          <gammavalue from='231' to='240'/>
          <gammavalue from='239' to='248'/>
          <gammavalue from='247' to='248'/>
          <gammavalue from='255' to='255'/>
      </gammalist>
```

MFFGOF

The filter MFFGOF is only designed as input filter and is reading two SAP formats, which are combined in the name **Generic Output Format (GOF)**:

- Output Text Format (OTF)
- ABAP List Format (ALF)

The GOF input filter identifies the format by the **`FORMAT`** parameter of the data stream header: OTF, if **`*FORMAT=OTF`**, and ALF, if **`*FORMAT=LIST`**.

OTF is a page-describing format. Therefore, it does not need any additional layout rules, the settings of the GOF profile do not influence the processing of OTF data streams. During reading and converting into the internal meta format, they are ignored. The settings of the MFFGOF profile are only effective to ALF lists and influence exclusively the presentation of their layout. An exception is the element **`<stringappendmode>`**.

<stringappendmode.../>

Element affects the method how two strings in sequential order are displayed, if the second string is not repositioned.

In the OTF data stream, two positioning commands are used: MT (Move-To) and ST (STring). The values of the attribute **`value`** are **`absolute`** and **`relative`**.

If **`absolute`** is specified, the positioning of the second string is absolute, i.e. the position of the second string is calculated from the position and the length of the first string. If relative is specified, the positioning is relative to the second string, i.e. it is appended.

Relative positioning is recommended. In the output formatted text strings can have a different length because of the output format fonts. Thus, if absolute positioning is used, text strings may overlay each other.

MFFGOF - User Defined Encoding

For detailed information see “SAPGOF Special Characters” on page C.1.

MFFGOF - Medium Definitions

The layout presentation is specified in the ALF data stream by the parameters ***PJFORM** or ***PJPAPER** in the header depending of the ALF version. The keywords assigned to these parameters are of descriptive nature like **X_PAPER, Z_PA01** or **LETTER** or they contain a line and column assignment in its keywords like **Z_65_132**, i.e. **65** lines and 132 columns.

The GOF filter reads this header entry and looks in the profile file **MFFGOF.PRO** for an entry in the element **<medium definition.../>** with the same value of the attribute name.

Example: Medium Definition in the MFFGOF profile

```
<mediumdefinition name="X_65_132"
formdef="F1A10111"
pagedef="mypage"
copygroup = "1"
fontnormid = "12" />
```

The medium definition contains the following assignments according to specifications for an AFP data stream:

- font definition (fontnormid),
- page definition (pagedef),
- form definition (formdef) and
- copy group (copygroup).

The assignments are specified in more detail in the GOF profile in associated elements that are described in the following.

If the GOF filter does not find a medium definition with the name that was read out from the header, it tries to read out a line and column definition out of the name and tries to find a corresponding page definition in the GOF profile, see “MFFGOF - Page Definitions” on page 3.17. If it finds a suitable definition, it uses the parameter values and Courier with font size 10pt as font. Otherwise, an error message is issued.

MFFGOF - Font Assignments

Example: Font definitions in the MFFGOF profile

```
<fontlist>

  <font family="COURIER" serifstyle="SERIF" spacing="MONOSPACED" id="1"
                                            pitch="12000">
    <face   weight="MEDIUM" width="NORMAL" style="UPRIGHT" size="10" />
  </font>

  <font family="COURIER" serifstyle="SERIF" spacing="MONOSPACED" id="12"
                                            pitch="15000">
    <face   weight="MEDIUM" width="NORMAL" style="UPRIGHT" size="8" />
  </font>

  <font family="COURIER" serifstyle="SERIF" spacing="MONOSPACED" id="13"
                                            pitch="12000">
    <face   weight="BOLD" width="NORMAL" style="UPRIGHT" size="10" />
  </font>

</fontlist>
```

<fontlist>

In element fonts are defined in a list. The element **`<font.../>`** specifies which fonts can be used by the GOF filter. The assignment is done by the value of the attribute **`fontnormid`** of the corresponding element **`<mediumdefinition .../>`** that has to match the value of the attribute **`id`** of an element **`<font .../>`** inside the font list. If no corresponding **`id`** value is found, the font with **`id`** value 1 inside the font list is used.

The font is defined by the element **`<font.../>`** as well as the element **`<face.../>`** and its attributes.

MFFGOF - Page Definitions

The page definitions in the **`<input>`** element of the GOF profile are specified by the elements:

```
<pagesize.../>,
<margins.../> and, where applicable
<linesperinch.../>,
<resolution> and
<rotation>.
```

Example: Page definitions in the MFFGOF profile

```
<pagedefinition name="A4_portrait">
  <papersize format = "A4" orientation = "portrait" />
  <margins left = "2 cm" right = "2 cm" top = "1 cm" bottom = "1 cm" />
  <linesperinch value  = "8" />
  <rotation value = "0" />
</pagedefinition>

<pagedefinition name="A4_landscape">
  <papersize format = "A4" orientation = "landscape" />
  <margins left = "2 cm" right = "2 cm" top = "1 cm" bottom = "1 cm" />
  <linesperinch value = "8" />
  <rotation value = "0" />
</pagedefinition>

<pagedefinition name="LETTER_portrait">
  <papersize format = "letter" orientation = "landscape" />
  <margins left = "1 in" right = "1 in" top = "2 in" bottom = "2 in" />
  <linesperinch value = "8" />
  <rotation value = "0" />
</pagedefinition>

<pagedefinition name="LETTER_landscape">
  <papersize format = "letter" orientation = "landscape" />
  <margins left = "2 cm" right = "2 cm" top = "1 cm" bottom "1 cm" />
  <linesperinch value = "8" />
  <rotation value = "0" />
</pagedefinition>

<pagedefinition name="mypage">
  <papersize width = "297 mm" height = "21 cm" />
  <margins left = "2 cm" top = "15 mm" />
  <linesperinch value = "8" />
  <resolution rx="1440" ry="1440" />
  <rotation value = "270" />
</pagedefinition>
```

<papersize.../>

Element defines the paper size by the attributes **`format`** and **`orientation`**, as shown in the upper definitions of this example, or by the attributes **`width`** and **`height`**, as shown in the last definition of the example. As value for the attribute **`format`** all names of page formats can be used that are listed in Appendix B: Page Formats on page 118. The values for the attribute **`orientation`** are **`landscape`** and **`portrait`**. For the values of the attributes **`width`** and **`height`**, unit abbreviations are listed in **Table 3: Unit Ab**breviations on page 90.

<margin.../>

Element defines page margins. The following attributes can be specified: The right margin (attribute **`right`**), the left margin (attribute **`left`**), the top margin (attribute **`top`**), and the bottom margin (attribute **`bottom`**). The same unit abbreviations as for the attributes **`width`** and **`height`** in the element **`<papersize.../>`** can be used for the margin attributes.

<linesperinch.../>

Element defines in the attribute **`value`** the number of lines per inch.

<resolution.../>

Element defines the resolution. The resolution in x-direction is defined by the value in the attribute **`rx`**, the resolution in y-direction is defined by the value in the attribute **`ry`**. Default unit is of the attribute value is a Twentieth of Inch Point, i.e., 1/1440 inch.

<rotation.../>

Element specifies the rotation of a document page. Valid values for the attribute **`value`** are **`0`**, **`90`**, **`180`**, and **`270`** specifying the number of degrees of the rotation.

If a medium definition for the data stream was found, the GOF filter identifies the page definition to be used by means of the **`name`** attribute in the element **`<pagedefinition>`** that has to correspond to the value of the **`pagedef`** attribute in the element **`<mediumdefinition.../>`**.

Unit Abbreviation	**Meaning**
mm	Millimeter
cm	Centimeter
dm	Decimeter
m	Meter
in	Inch
inch	Inch
pt	Point (1/72 Inch)
tw	Twentieth of Inch Point (1/1440 Inch)
tcm	1/1000 cm

Note: Units can only be used in combination with integer values of the attributes!

MFFGOF - Form Definitions

The form definition (attribute **formdef**) in the **<mediumdefinition.../>** element allows the use of formdefs like in the case of AFP. With formdefs forms or overlays can be defined for ALF lists. The value of the attribute **formdef** specifies the name of a formdef object. This object has to be accessible directly for the filter. The drive and path of the object will be read by the filter in the resource list of the GOF profile as long as it is not specified fully qualified.

MFFGOF - Copy Group Definitions

The formdef consists of one or more copy groups (medium maps). Each of these copy groups can contain definitions for overlays (also different overlays for different pages), tray assignment and simplex/duplex printing. The value of the attribute **copygroup** specifies which of the copy groups inside the formdef shall be used for the ALF list. Currently, the GOF filter does not interpret this value: It uses the first copy group anyway.

MFFGOF - Page Orientation

<autorotate.../>

Element controls page orientation from landscape to portrait.

Note: **<autorotate>** is used, if no tray name is specified as medium map name in elements like **<pagedefinitions>**, **<mediumdefinitions>**, with the SAPGOF command OP (OPEN PAGE), and if the printer cannot print landscape jobs.

Document pages are generated with orientation landscape, i.e., in the data stream pages have the landscape orientation. Using **<autorotate>** pages can be rotated to portrait on above mentioned conditions, i.e. they are rotated 90 degrees counterclockwise.

The values of the attribute **value** are **false** and **true**. If the value **false** is specified, the page rotation is not active. If the value **true** is specified, pages are rotated.

MFFGOF - Resource Definitions

For more details, see “MFFAFP - Resources” on page 3.8.

MFFGOF - Trays

<tray.../>

Element controls the definition of paper trays. The value of the attribute **name** is the name of the paper tray. The name must correspond with the value that is specified with the attribute **name** of the subelement **<inputtray>** of the element **<inputtray>**. For more information, see “MFFAFP - Trays” on page 3.9.

MFFGOF - Barcode Definitions

For each barcode name used in OTF a corresponding barcode type can be specified as well as the fact whether a check digit has to be generated or not. Furthermore it can be specified whether the content of the barcode has to be written as text under the symbol and the font to be used for it (the so-called .Human Readable Information., HRI). Additionally the width can be defined that has to be used for the symbol depending of the information contained in the barcode, as well as the decision whether the maximum width predetermined in the OTF file has to be considered or not.

<barcode.../>

Within the element **`<barcodelist>`** the element defines the barcodes that are to be used. The barcode name has to match with the barcode name in the OTF file (attribute **`name`**). The barcode type (attribute **`type`**) can be one of the types listed in **Appendix D: Barcodes on page 128**. The number of the check digits to be generated is identified by the attribute **`checkdigits`**.

The attribute **`displayhri`** specifies whether a text with the content of the barcode shall be generated in addition to the barcode symbol.

The width of the barcode symbol can be controlled by the attribute **`modulewidth`**. It specifies how wide a module (= a thin bar) has to be presented. With this attribute, the same units can be used as in the page size attributes.

The attribute **`respectmaxwidth`** specifies whether the value of the maximum barcode width specified in the OTF file has to be used or not.

The attribute **`fontid`** specifies the font to be used in the case **`displayhri`** is set to **`TRUE`** **(**-> **`fontlist`**).

Example: Barcode definition in the MFFGOF profile

```
<barcodelist>
  <barcode name="BC_CD39C" type="Code 39"
                           checkdigits='1'
                           displayhri="FALSE"
                           modulewidth="18 tw"
                           respectmaxwidth="FALSE"
                           fontid='13' />
  <barcode name="ARTNR"    type="Code 128"
                           checkdigits='0'
                           displayhri="TRUE"
                           modulewidth="20 tw"
                           respectmaxwidth="FALSE"
                           fontid='13' />
  <barcode name="MBBARC1"  type="EAN 8"
                           checkdigits='0'
                           displayhri="TRUE"
                           modulewidth="20 tw"
                           respectmaxwidth="FALSE"
                           fontid='14' />
</barcodelist>
```

MFFGOF - Colors

Example: Highlight Color (Spot Color) List in the MFFGOF Profile

```
<highlightcolorlist>
   <highlightcolor name="COL0N" foreground-rgb="#00000000"
                                background-rgb="#00808080" />
   <highlightcolor name="COL0H" foreground-rgb="#000000FF"
                                background-rgb="#00808080" />
   <highlightcolor name="COL0V" foreground-rgb="#00808080"
                                background-rgb="#000000FF" />
   <highlightcolor name="COL1N" foreground-rgb="#00000000"
                                background-rgb="#008585A6" />
   <highlightcolor name="COL1H" foreground-rgb="#00000000"
                                background-rgb="#000000FF" />
   <highlightcolor name="COL1V" foreground-rgb="#000000FF"
                                background-rgb="#00808080" />
      .
      .
      .
   <highlightcolor name="COL7N" foreground-rgb="#00000000"
                                background-rgb="#00A185A6" />
   <highlightcolor name="COL7H" foreground-rgb="#00000000"
                                background-rgb="#00E06090" />
   <highlightcolor name="COL7V" foreground-rgb="#00E06090"
                                background-rgb="#00808080" />
```

<highlightcolorlist.../>

The subelement **`<highlightcolor>`** of **`<highlightcolorlist>`** with its attributes **`name`**, **`foreground-rgb`** and **`background-rgb`** defines a highlight color.

The value of the attribute name specified a predefined SAPGOF color. For a set of colors, see profile file **MFFGOF.PRO**.

Note: The SAPGOF command PC (PRINTCONTROL) and the name of printcontrol, e.g., **`PCCOL0N`** for a color, defines that a color is assigned to the data following the command.

If the attribute **`foreground-rgb`** is specified, the foreground color, i.e., line or font color, is defined. The value **`PCCOL0N`** specified the default font color.

If the attribute **`background-rgb`** is specified, the background color is defined. The value **`PCCOL0H`** specified the default background color.

If no color list is defined in the profile, the default is black and white.

MFFGOF - Printing

<duplexcontrol.../>

Element controls simplex and duplex printing. The values of the attribute **`value`** are **`simplex`**, **`duplexfront`** and **`duplexback`**.

If the attribute **`simplex`** is specified, data are printed on one side of a page.

If the attribute **`duplexfront`** is specified, printing starts on the front side of a page and is continued on the backside.

If the attribute **`duplexback`** is specified, printing starts on the backside of a page.

Conversion of OTF to AFP with cpmcopy

If the attribute **`PJFORM`** in the header of the OTF file is specified, the name of a formdef is found via the medium definition, see "MFFGOF - Medium Definitions" on page 3.15 and "MFFGOF - Form Definitions" on page 3.19**.** The formdef name can be submitted for the MFFAFP filter with the command line option **`-copyformdef`**. Additionally, the command line option **`-selfcontained`** must be specified in order to include the formdef with the specified name in the AFP output profile.

In the MFFAFP profile file the attribute value of **`<usetrayasmediummap>`** must be **`true`** to take the name of the copy group that is to be referenced from the tray name of the OP (Open Page) command in the OTF file. If the tray name is omitted, the element **`<autorotate>`** in the MFFGOF profile file can be used to place the page correctly without a copy group on a medium in portrait orientation, see "MFFGOF - Page Orientation" on page 3.19.

MFFIJP

Note: It is recommended to generate separate profiles for spot color and full color mode.

Sample profile file names:

- MFFIJP_SPOTCOLOR.PRO (spot color)
- MFFIJP_CMYK.PRO (full color)

MFFIJP - Global Settings (Spot Color)

In the following global settings are described, which overwrite the predefined filter settings.

Example: Global settings in the MFFAFP profile

```
<warninglevel   value="2"      />
<rastersettype  value="screen" />
```

<warninglevel.../>

Element specifies the warning type for warning messages issued during the process run. The value of the attribute **`value`** specifies the warning level.

The default value is **`2`**. Levels 0 to 3 are available.

<rastersettype.../>

Element is used to provide a specified area with a defined pattern type. The values of the attribute **`value`** are **`screen`**, **`screenmodified`**, **`afp`**, **`afpmodified`**, **`roughw0, roughw15, roughw45, roughw75, fine0, fine15, fine45`** and **`fine75`**.

MFFPCL

MFFPCL - Global Settings

In the following global settings are described, which overwrite the predefined filter settings.

Example: Global Settings in the MFFPCL profile

```
<warninglevel   value="2"      />
```

<warninglevel.../>

Element specifies the warning type for warning messages issued during the process run. The value of the attribute value specifies the warning level. The default value is 2. Levels 0 to 3 are available.

MFFPCL - Code Pages

Example: Code Page Definition in the MFFPCL profile

```
<codepagelist>
  <codepage name="12U" iana="IBM850"/>
</codepagelist>
```

<codepagelist>

The subelement <codepage.../> of the element <codepagelist...> defines several code pages.

MFFPCL - Fonts

Example: Font Definition in the MFFPCL profile

```
<font family="Courier">
  <face weight="MEDIUM" style="UPRIGHT">
     <raster fontfile="DATAM4" fontfiletype="PCL" size="1.599999"
                                            fixedsymbolset="TRUE"/>
  </face>
 <face weight="BOLD" style="UPRIGHT"  fontselector="0p0s3b4102T"
                                      embed="NEVER" />
</font>
</fontlist>
```

MFFPCL - Trays

For additional information, see “MFFAFP - Trays” on page 3.9.

Example: Tray Definition in the MFFPCL profile

```
<traylist>
<!-- = inputtray =============================================== -->
  <inputtray deviceid=.1. name=.UpperTray.  mediatpyeid0=.2./>
  <inputtray deviceid=.2. name=.ManualFeed.      />
  <inputtray deviceid=.3. name=.ManualEnvelope.  />
  <inputtray deviceid=.4. name=.LowerTray.       />
  <inputtray deviceid=.5. name=.LargeCapacity.   />
  <inputtray deviceid=.6. name=.Envelope.        />
  <inputtray deviceid=.7. name=.Automatic.       />
</traylist>
```

<inputtray.../>

Element <traylist> as subelement of <inputtray.../> specifies the tray definition for input and output. The attributes are deviceid, name and mediatypeid. The value of the attribute mediatypeid specifies the printing medium:

0 plain paper

1 bond paper

2 special paper

3 glossy film

4 transparency film

MFFPCL - Color Profile

Example: Definition of a Color Profile List in the MFFPCL profile

```
<colorprofilelist>
   <!-- <colorprofile name="GenericRGB"  /> -->
   <!-- <colorprofile name="GenericCMYK" /> -->
   <!-- <colorprofile name="GenericGray" /> -->
</colorprofilelist>
```

<colorprofile.../>

Element specifies the name of the color profile. The values of the attribute name are GenericRGB, GenericCMYK and GenericGray

Note: The list can contain any number of profiles. Just the current profile should be uncommented.

MFFPCL - Input

In the following default settings are described, which overwrite the predefined filter settings. All elements are subelements of <input>.

Example: Default Settings in the MFFPCL profile

```
<defaultsetting>
   <pagesize value="26"/>
   <pagelength value="68"/>
   <imageresolution value="300"/>
   <symbolset value="341"/>
</defaultsetting>
```

The following subelements of <defaultsetting> are available:

<pagesize.../>

Element defines the paper size. The value of the attribute value specifies the paper size. The supported page sizes and corresponding values are listed in Table 4: Paper Sizes on page 103.

<pagelength.../>

Element controls the form length. The value of the attribute value specifies the number of lines.

<imageresolution.../>

Element controls the resolution. The value of the attribute value specifies the resolution.

<symbolset.../>

Element controls the symbol set. The value of the attribute value specifies a symbol set. The default symbol set is PC-8.

The various symbol sets include symbols and characters as well as language dependent special characters, graphic characters, scientific symbols, etc. The supported symbol sets and related values are listed in Table 5: Symbol Sets on page 104.

Paper Sizes

Format Name	Value	Format Name	Value
A3	27	**Executive**	1
A4	26	**Hagaki Postcard**	71
A5	25	**JIS B4 Pager**	46
Envelope #10	81	**JIS B5 Paper**	45
Envelope B5	100	**Ledger**	6
Envelope C5	91	**Legal**	3
Envelope DL	90	**Letter**	2
Envelope Monarch	80	**Oufuku-Hagaki Postcard**	72

Symbol Sets

Name	Value	Name	Value
Desktop	234	**MC-Text**	394
Iso L1	14	**PC-1004**	298
Iso L2	78	**PC-775**	853
Iso L5	174	**PC-8**	341
Iso L6	206	**PC-8 DN**	373
Iso L9	302	**PC-8 TK**	308
Iso-11	19	**PC-850**	405
Iso-15	9	**PC-852**	565
Iso-17	83	**PS Text**	330
Iso-21	39	**Roman-8**	277
Iso-4	37	**Win 3.0**	309
Iso-6	21	**Win Balt**	620
Iso-60	4	**Win L1**	629
Iso-69	38	**Win L2**	293
Legal	53	**Win L5**	180

Example: Overlay Definition in the MFFPCL profile

```
<macrolist>
    <file name="EDI_4035M.pcl" >
       <macro name="Overlay"/>
    </file>
</macrolist>
```

<macrolist.../>

Element controls the access of a PCL file to an external macro file to integrate an overlay. The subelement <file> of the element <macrolist> specifies the path of the macro file.

MFFPCL - Output

In the following default settings are described, which overwrite the predefined filter settings. All elements are subelements of **`<output>`**.

<nohpgl.../>

Element controls the generation of HPGL (Hewlett Packard Graphics Language). The values **`true`** and **`false`** of the attribute **`value`** specify the HPCL support of the PCL printer. The default value is **`false`**.

<color.../>

Element specifies whether the hardware device in a color printer or output file is a color-printing file. The values of the attribute **`value`** are **`true`** and **`false`**. The default value is **`false`**.

<fullprintablearea.../>

Element controls with the values **`true`** and **`false`** of the attribute **`value`**, whether the printer can use the complete physical page (edge-to-edge printing).

MFFPCL - Resources

Example: Resource Definition in the MFFPCL profile

```
<resourcelist>
   <files   path="resource" type="PCL" extension="FLJ"/>
   <files   path="resource" type="PCL" extension="ovl"/>
   <files   path="\resource" type="ICC" extension="icc"/>
   <files   path="\resource" type="ICC" extension="icm"/>
</resourcelist>
```

<resourcelist>

With subelement **`<files.../>`** all resources are specified. The value of the attribute `path` contains the path information for the resources. The values of the attribute `type` can be **`PCL`**, **`TrueType`**, **`Type1Font`** or Type1Metrics.

MFFPDF

MFFPDF - Fonts

Example: Font definitions in the MFFPDF profile

```
<fontlist>
  <font family="Gothic" serifstyle="SERIF" spacing="MONOSPACED" >
    <face weight="MEDIUM" width="NORMAL"
                          style="UPRIGHT" devname="Gothic"/>
  </font>

  <font family="Univers CE" reftype="TrueType">
    <face weight="MEDIUM" style="UPRIGHT" devname="Univers"
                          fontfile="Univer"   fontfiletype="TrueType"
                          embed="always"/>
    <face weight="BOLD"   style="UPRIGHT" devname="Univers-Bold"
                        fontfile="Univerb"  fontfiletype="TrueType"/>
  </font>
</fontlist>
```

<fontlist>

Element specifies for which font name in the Presentation Area which font for writing PDF files should be used. In PDF files fonts can be referenced as well as completely included.

If a TrueType font is to be referenced or included, the attribute **`reftype`** with the value **`TrueType`** has to be specified in the element **`<font>`**. Otherwise, a reference to a Type 1 font is written.

If the attribute **`fontfiletype with the value TrueType`** is specified for the element **`<face>`**, it is searched for the font with extension **`ttf`** set by **`fontfile`**, otherwise **`pfb`** (for Type 1). The font name can be specified as absolute name (with path). Otherwise, it is searched in the current directory or in one of the paths, which are specified as described in “MFFPDF - Resources” on page 3.29.

MFFPDF - Resources

<resourcelist>

Element defined the font location. If font files are not specified with an absolute path in the element `<fontlist>`, paths can be defined in the element `<resourcelist>`.

Attention: When search paths are specified, the current directory is not scanned

Example: Resource definition in the MFFPDF profile

```
<resourcelist>
   <files   path="c:\winnt\fonts"                  extension="ttf" />
   <files   path="\\fileserver\drive\fonts\type1" extension="pfb" />
</resourcelist>
```

MFFPDF - Output

In the following default settings are described, which overwrite the predefined filter settings. All elements are subelements of **`<output>`**.

<replacepattern>

Element controls the use of fill patterns for rectangular areas. Fill patterns can be placed by the appropriate gray values. The values of the attribute **`type`** are **`false`** and **`true`**.

If **`true`** is specified, the fill pattern is replaced by the gray value. If **`false`** is specified, the fill pattern is used. The default value is **`false`** is specified, the fill pattern is replaced by the gray value. A replacement reduces the file size and increase the writing performance.

<jpegquality.../>

Element controls visual integrity. The value of the attribute **`value`** contains a percentage number. The default value is **`75`**. Note: Visual integrity improves at the expense of the file size.

<fonthandling>

Element controls the handling of fonts that are not defined in the element **`<font>`**, see "MFFPDF - Fonts" on page 3.29. The values of the attribute **`value`** are **`auto`**, **`rasterize`** and **`converttotype3`**.

If **`auto`** is specified, the default fonts are used. If **`rasterize`** is used, texts are generated as bitmaps. If **`converttotype3`** is used, the fonts of the input filter are converted to Adobe PostScript Type-3 fonts.

<writearchivefields>

Element controls whether archive fields are written as annotations. The values of the attribute **`value`** are **`not`** and **`annotations`**. If **`not`** is specified, writing of archive fields is ignored. If **`annotations`** is specified, archive fields (AFP TLEs and index values) are written as PDF annotations (comments).

Note: In the data stream, Acrobat comments cannot be assigned to a specified page. Whereas annotations generated with **`<writearchivefields>`** can be assigned to specific pages.

Section 4
Messages

Syntax

This section describes the messages that are written to the console and the log files.

The following exemplary message shows the general structure with and without filename:

```
ABC1234I: This is a text with 'Parameter'
XYZ4321E myfile.xml(12:34): This is another text
```

Each message starts with a triple-digit component ID (e.g., ABC) followed by a message number (e.g., 1234). The characters used have the following meaning:

F - Fatal error. Processing will be cancelled.

S - Serious error in one component; the result is probably unusable. Processing will not be cancelled.

E - Error that, in all probability, leads to an undesirable result. Processing will be continued.

W - Warning that may lead to an undesirable result. Processing will continue.

I - Information describing the process.

V - Verbose information describing the process. These messages cannot be written by all programs. They usually can be controlled by parameter.

The file name will be written if the error is related to a file (myfile.xml). After the file name, the position inside the file can be written in brackets:

- The first number (e.g., 12) identifies the row; the second number (e.g., 34) the column.
- The column number may also be omitted.

The specification of row and column is written as closely as possible. But the message can also appear before. If an error number is related to a binary file, the number corresponds to the offset inside the file.

Inside the message text, variable parts (parameter) can appear that are often written in single or double quotation marks.

In batch programs, messages are usually written into the console and simultaneously in a log file. In programs with a graphical user interface, the messages can be written in a protocol window. Depending on the program, the control of it is presented in different ways.

Messages

AFP - Advanced Function Printing

AFP1011F The FORMDEF ‘STRING’ could not be found.

Explanation: The specified FORMDEF could not be found. Therefore any further request to a copy group (medium map) could not be resolved. The search order for FORMDEFs is:

- Inline resources.
- External resources as specified in the MFFAFP profile.
- Single objects in the file system, with passes specified in the MFFAFP profile.

Action: Check the name of the FORMDEF specified in the command line or the application profile. Make sure that the search passes defined in the MFFAFP profile are set properly.

AFP1025F Invalid subtype 'STRING', DOC, DOCUMENT, OVL, OVERLAY, PSG or PAGESEG expected

Explanation: When writing an AFP file, the type of the file could be specified on the command line or the application profile, like afp.pageseg for an AFP page segment. If no subtype is specified, document is used as the default. The type of an AFP file can be:

- A regular AFP document (subtype DOC or DOCUMENT).
- An overlay (subtype OVL or OVERLAY).
- A page segment (subtype PSG or PAGESEG).

All other subtypes are illegal.

Action: Check the specified subtype.

AFP1026F Page segments and overlays cannot span multiple pages

Explanation: If the AFP file to write is a page segment or an overlay, it can only store one page. Only AFP documents can store multiple pages.

Action: Make sure only one page is copied to a page segment or overlay.

AFP1024E Default character CHAR (STRING) not found in any code page

Explanation: The default character that can be specified in the MFFAFP profile is used whenever a character cannot be resolved in one of the specified code pages. The worst case in that scenario is when the default character itself cannot be found in any of the specified code pages. The internal character representation is in Unicode. If a character is written to the AFP output file, the Unicode character is first translated into an IBM character name and all given code pages are checked to resolve that character.

Action: Check the default character in the MFFAFP profile.

AFP1031E IBM character name in codepage 'CODEPAGE' (codepoint NUMBER) is empty

Explanation: The character name within the AFP codepage is empty (only space characters or null characters) and cannot be used. A character name must have at least one valid character.

Action: Check the codepage object.

AFP1032E Error copying temporary file 'FILENAME' (RC

Explanation: When inline resources (self-contained files) are requested, the inline resource file is generated as a temporary file along with document that is to be written. As inline resources must reside at the beginning of the AFP document, the two temporary files are combined into one file, when the output file is closed. This requires twice the disk or file system space for the final file as the temporary files can only be deleted when the copy process is finished.

Action: Check the disk or file system conditions and make sure that there is enough space for the temporary files and the resulting file. Check the given return code.

AFP1034E Illegal or unsupported codepage IANA name 'STRING' in profile, assuming 'STRING'

Explanation: When a codepage is requested, an attempt is made to find the AFP codepage object in the inline or external resources or as a single object in the file system. If the codepage could not be found, a substitute codepage can be specified in the profile. This codepage is specified by its IANA (Internet Assigned Numbers Authority) name. A number of built in code pages are supported, but the given name is unknown.

Action: Check the given IANA name.

AFP1042F Device name not set in profile (family='FONTFAMILY')

Explanation: The device name for the requested font family was not specified in the MFFAFP profile. A default AFP core font like C0H..... (Helvetica) for sans serif fonts will be used. The device name of a font is the 8 character object name that contains the character set.

Action: Check your MFFAFP profile.

AFP1049E Error reading IOCA image (RC)

Explanation: A general error occurred while reading an IOCA image, that is an imbedded object, within the AFP data stream.

Action: Check the return code.

AFP1053E Error reading font 'OBJECT', font has no font pattern map

Explanation: The AFP font file does not contain a valid font pattern map and can therefore not be used.

Action: Check the given font object.

AFP1055E Error writing IOCA image: 'STRING' (RC)

Explanation: A general error occurred while writing an IOCA image to the output AFP file. This might have multiple reasons:

- A write error occurred.
- The disk or file system is full.

Action: Check the output file and the given return code.

AFP1068E Error reading resource 'OBJECT' from file 'FILENAME' ('RC')

Explanation: An error occurred while reading a single object from an inline resource library or an external resource library.

Action: Check the resource library.

AFP1006W The overlay 'OBJECT' could not be found

Explanation: The given overlay could not be found and will be ignored. The search order for overlays is:

- Inline resources.
- External resources as specified in the MFFAFP profile.
- Single objects in the file system, with passes specified in the MFFAFP profile.

Action: Check the overlay name and make sure, that the search passes defined in the MFFAFP profile are set properly.

AFP1007W The page segment 'OBJECT' could not be found

Explanation: The given page segment could not be found and will be ignored. The search order for page segments is:

- Inline resources.
- External resources as specified in the MFFAFP profile.
- Single objects in the file system, with passes specified in the MFFAFP profile.

Action: Check the page segment name and make sure, that the search passes defined in the MFFAFP profile are set properly.

AFP1008W The coded font 'OBJECT' could not be found

Explanation: The given coded font could not be found and will be ignored. The search order for coded fonts is:

- Inline resources.
- External resources as specified in the MFFAFP profile.
- Single objects in the file system, with passes specified in the MFFAFP profile.

Action: Check the coded font name and make sure, that the search passes defined in the MFFAFP profile are set properly.

AFP1009W The code page 'OBJECT' could not be found

Explanation: The given code page could not be found and will be ignored. The search order for code pages is:

- Inline resources.
- External resources as specified in the MFFAFP profile.
- Single objects in the file system, with passes specified in the MFFAFP profile.

If the code page is not found as an AFP resource, the code pages defined by IANA names in the profiles are used. If it is still not found in the profile a default codepage is used.

Action: Check the code page name and make sure, that the search passes defined in the MFFAFP profile are set properly.

AFP1010W The character set 'OBJECT' could not be found

Explanation: The given character set could not be found and will be ignored. The search order for character sets is:

- Inline resources.
- External resources as specified in the MFFAFP profile.
- Single objects in the file system, with passes specified in the MFFAFP profile.

Action: Check the character set name and make sure, that the search passes defined in the MFFAFP profile are set properly.

AFP1012W The container object 'OBJECT' could not be found

Explanation: The given object stored in an object container or as an external file could not be found and will be ignored. The search order for objects is:

- Inline resources.
- External resources as specified in the MFFAFP profile.
- Single objects in the file system, with passes specified in the MFFAFP profile.

Action: Check the object set name and make sure, that the search passes defined in the MFFAFP profile are set properly.

AFP1019W Font 'FONT', size=SIZE, weight='WEIGHT', style='STYLE' not found, will be replaced with standard 'FONT'

Explanation: The given font could not be found. The generic font name must be mapped to an AFP device name. There is not entry in the MFFAFP profile to map the generic font name to the AFP device name and thus a default core font is used.

Action: Check the font entries in the MFFAFP profile.

AFP1020W Font 'FONT' replaced with standard 'FONT'

Explanation: The given font could not be resolved and is therefore mapped to a default font, using the font attributes SERIF/SANSERFIF and MONOSPACED/ PROPORTIONAL. The default core fonts are:

- Courier
- Helvetica
- Times Roman

Action: Check the font entries in the MFFAFP profile.

AFP1021W Character CHAR (STRING) not found in any code page, replaced with CHAR

Explanation: All text is internally stored in Unicode. If a text item is generated by the AFP output driver, it is translated into a codepage/character set representation. Therefore the Unicode character is searched in all available codepages. None of the given codepages contains the character to be generated.

Action: Check the codepage list in the MFFAFP profile

ARG - Command Line Parsing

ARG0200E Response file 'FILENAME' could not be opened

Explanation: A command line file (response file) can be given indicated by an argument leading '@'. The lines of the response file are inserted at the location of the command line. At any place a line feed can be announced, where space characters are allowed. The following example:

```
myprog -opt @myparms.rsp -otheropt
```

with content of the file **myparms.rsp**

```
-input in.txt
-output out.txt
```

gives the same result as the following command line:

```
myprog -opt -input in.txt -output out.txt -otheropt
```

The error message is displayed, if the response file can't be opened.

Action: Double check the presence and rights of the given file.

ARG0201E Response file line is too long

Explanation: A command line file (response file) line may contain at most 256 characters.

Action: Shorten the lines, for example by insertion of a line feed at a space.

ARG0202E Too many nested response files, cannot process 'FILENAME'

Explanation: Command line files (response files) may invoke other response files, but only up to a (nesting) depth of 6. A likely error is recursion, i.e., a response file invokes itself.

Action: Double check your command line.

ARG0203E Label 'LABEL' not found

Explanation: Some applications allow jumping to a label in a command line file. A label is designated by a leading ':' before the label name. That label doesn't exist in the response files.

Action: Double check your command line.

ARG0204E Empty option 'CHAR' is not allowed

Explanation: The option character '-' (on MS Windows and OS/2 also '/') has to be followed by a letter.

Action: Double check your command line.

ARG0205E Illegal value 'VALUE' for option 'OPTION'

Explanation: The value is not valid for the option.

Action: Double check your command line.

ARG0206E Option 'OPTION' is not defined

Explanation: Unsupported option encountered.

Action: Double check your command line.

ARG0207E Option 'OPTION' is ambiguous, be more specific

Explanation: Options may be abbreviated, if the shortage is ambiguous. For example the option **namei** is a valid abbreviation, because it is definitely different from **nameout**. But **name** is ambiguous, because both **namein** and **nameout** start with it.

Action: Double check your command line.

ARG0208E Value 'VALUE' is not expected for option 'OPTION'

Explanation: This value isn't supported for this option.

Action: Double check your command line.

ARG0209E Option 'OPTION' requires a value

Explanation: Some command line options require another value like **- input myfile.txt**. If that value is missing, this error message is displayed.

Action: Double check your command line.

ARG0210E Illegal numeric value 'VALUE' for option 'OPTION'

Explanation: The given option requires a numerical value, i.e., only digits are accepted.

Action: Double check your command line.

ARG0211E Illegal switch value 'SWITCH' for option 'OPTION'

Explanation: The given option is a switch. Switches can be turned on ('+') or off ('-'). If no '+' or '-' is used, generally the switch is turned on. Other Values aren't allowed for this kind of options.

Action: Double check your command line.

ARM - AFP Resource Manager

ARM1000F Error initializing argument parser

Explanation: While initializing the command line parser an error occurred. Processing is terminated.

Action: Contact LRS technical support staff.

ARM1001F Error allocating instance data

Explanation: During internal initializing the Resource Manager instance an error occurred.

Action: Processing is terminated.

ARM1002F Error allocating structured field data

Explanation: During internal building the buffer for AFP record data an error occurred. Processing is terminated.

Action: Contact LRS technical support staff.

ARM1003F Error allocating resource reference

Explanation: During internal building the buffer of a resource reference an error occurred. Processing is terminated.

Action: Contact LRS technical support staff.

ARM1004F Error adding resource reference (RC)

Explanation: During internal enhancing the list of resource references an error occurred. Processing is terminated.

Action: Contact LRS technical support staff.

ARM1005F Error adding path entry 'ENTRY' (RC)

Explanation: During internal enhancing the list of path entries an error occurred. Processing is terminated.

Action: Contact LRS technical support staff.

ARM1006F Error creating I/O streams (RC)

Explanation: During internal initializing an extended IO object an error occurred. Processing is terminated.

Action: Contact LRS technical support staff.

ARM1007F Error opening document file 'FILE' (RC)

Explanation: The file used as AFP document to create the resource reference list cannot be opened. Following reasons are possible:

- The file doesn't exist.
- You have no read access to the drive/file system/directory.
- The given file is locked by another application.

Action: Check the given file.

ARM1008F Error opening resource object file 'FILE' (RC)

Explanation: The given file with a single AFP resource cannot be opened. Following reasons are possible:

- The file doesn't exist.
- You have no read access to the drive/file system/directory.
- The given file is locked by another application.

Action: Check the resource object file.

ARM1009F Error opening resource file 'FILE' for reading (RC)

Explanation: The given resource file (containing several single resources) cannot be opened. Following reasons are possible:

- The file doesn't exist.
- You have no read access to the drive/file system/directory.
- The given file is locked by another application.

Action: Check the resource library.

ARM1010F Error opening resource file 'FILE' for writing (RC)

Explanation: The given resource file which must be regenerated cannot be opened. Following reasons are possible:

- The file already exists with write protection.
- You have no read access to the drive/file system/directory.
- The given file is locked by another application.

Action: If necessary, delete the file that must be regenerated.

ARM1011F Error writing structured field record (RC)

Explanation: During writing an AFP record (structured field) an error occurred. Probable reason: disk is full.

Action: Check the given output file.

ARM1012F Error copying structured field record (RC)

Explanation: During copying an AFP record (structured field) an error occurred. Probable reason: disk is full.

Action: Check the given output file.

ARM1013F Data stream byte 0x5A expected

Explanation: During reading an AFP file or a resource (exception is a MO:DCA file) data stream byte with value 0x5A is expected to precede each record. The byte was not found. Reading of input file is terminated.

Action: Check input file issued in the error message. The value given in brackets after the file name indicates the offset within the file.

ARM1014F Structured field must begin with 0xD3

Explanation: AFP data must be provided with a structured field per record. It always has the identifier 0xD3. The identifier was not found at the expected position. Reading of output file is terminated.

Action: Check input file issued in error message. The value given in brackets after the file name indicates the offset within the file.

ARM1015F Error reading structured field header (RC)

Explanation: All the records of an AFP file start with an 8-byte header (beside 0x5A). During reading the header an error occurred. Reading of input file is terminated.

Action: Check input file issued in error message. The value given in brackets after the file name indicates the offset within the file.

ARM1016F Structured field size SIZE exceeds 32K

Explanation: AFP records have a maximum length of 32K (32768). A longer record is found. Reading of input file is terminated.

Action: Check input file issued in error message. The value given in brackets after the file name indicates the offset within the file.

ARM1017F Error reading structured field data (RC)

Explanation: Header data of an AFP record are followed by record related data. Reading data an error occurred. Reading of input file is terminated.

Action: Check input file issued in error message. The value given in brackets after the file name indicates the offset within the file.

ARM1018F Error reading data stream byte 0x5A (RC)

Explanation: Each AFP record starts with a data stream byte with the value 0x5A. During reading the byte an error occurred. Reading of input file is terminated.

Action: Check input file issued in error message. The value given in brackets after the file name indicates the offset within the file.

ARM1019F Only COUNT libraries can be specified, aborted

Explanation: The number of resource libraries that can be specified on the command line is limited (static list) and was exceeded. For safety reasons processing is terminated.

Action: Check the command line.

ARM2000E Name length LENGTH must be more than 8 bytes

Explanation: Name fields in AFP records must have a length of 8 bytes. According to specified record length remaining name length is shorter than 8 bytes.

Action: Check input file issued in error message. The value given in brackets after the file name indicates the offset within the file.

ARM2001E Illegal name character CHAR

Explanation: AFP object names may consist of predefined characters. The given character is not valid. If objects are referenced, they may not be found.

Action: Check input file issued in error message. The value given in brackets after the file name indicates the offset within the file.

ARM2002E CFI structured filed length LENGTH must be more than 16

Explanation: Coded Font Index (CFI) record data must have a length of 16 bytes. The given length is not correct. The record cannot be interpreted.

Action: Check input file issued in error message. The value given in brackets after the file name indicates the offset within the file.

ARM2003E BMM structured filed length LENGTH must be more than 8

Explanation: Data of a Begin Medium Map (BMM) record must have a length of at least 8 bytes. The given length is not correct. The record cannot be interpreted.

Action: Check input file issued in error message. The value given in brackets after the file name indicates the offset within the file.

ARM2004E FQN (0x02) triplet length LENGTH must be more than 10 bytes

Explanation: A Fully Qualified Name (FQN) triplet must have a length of at least 10 bytes. The given length is not correct.

Action: Check input file issued in error message. The value given in brackets after the file name indicates the offset within the file.

ARM2005E Cannot resolve TYPE 'NAME'

Explanation: The specified type of resource object cannot be resolved. The object could not be found as single resource or not found in the given resource libraries. Output file to be generated is incomplete.

Action: If necessary, check the specified paths and file extensions for single resources in the command line.

ARM2006E Cannot resolve medium map 'NAME' in FORMDEF 'FORMDEF'

Explanation: A medium map does not exist as single resource, but it is part of a FORMDEF (copy group). The given medium map cannot be found in the specified FORMDEF. Thus, output file to be generated is incomplete.

Action: Check FORMDEF specified in the command line.

ARM2007E Medium map 'NAME' cannot be resolved without a FORMDEF

Explanation: Medium maps are not available as single resources, but they are part of a FORMDEF (copy group). Thus, the FORMDEF must be specified (if resource libraries are not specified). This is not true. So, medium maps cannot be found. The output file to be generated is incomplete.

Action: Specify the FORMDEF that must be used during printing on the command line.

ARM2008E Medium map 'NAME' not found in FORMDEF 'FORMDEF'

Explanation: A medium map does not exist as single resource, but it is part of a FORMDEF (copy group) that is searched in resource libraries. The FORMDEF could not be found in any library. The output file to be generated is incomplete.

Action: Check the specified resource libraries.

ARM2009E Resource name 'NAME' truncated to 8 characters

Explanation: AFP object names have a maximum length of 8 characters. If specifying longer names (for example on the command line) the name is truncated to 8 characters.

Action: Check your specifications on the command line.

ARM3000W Argument 'ARGUMENT' without option ignored

Explanation: Input on the command line must be specified as option (e.g., afprm -FORMDEF=F1TEST). Parameters without preceding option are not allowed and ignored (e.g., afprm F1TEST).

Action: Check your specifications on the command line.

ARM3001W Only one FORMDEF supported, parameter ignored

Explanation: Running current version of the Resource Manager specification of only one FORMDEF per job is allowed. The specification of multiple FORMDEFs is ignored.

Action: Check your specifications on the command line.

ARM3002W Duplicate output file name, previous name 'FILE' ignored

Explanation: The name of resource library to generated can be specified only once on the command line. If the name is specified more than once, the one entered last is used.

Action: Check your specifications on the command line.

ARM3003W Illegal exclude character 'CHAR' ignored

Explanation: Using exclude option excluding certain resource types during output generation the following characters (upper/lower case) are allowed:

- 'C' for Character Set.
- 'F' for Form Definition.
- 'O' for Overlay.
- 'S' for Page Segment.
- 'T' for Codepage.
- 'X' for Coded Font.

Other characters are ignored.

Action: Check the specifications on the command line.

ARM3004W Illegal ignore character 'CHAR' ignored

Explanation: Using the ignore option (ignoring resource types during input), the following characters (upper/lower case) are allowed:

- 'C' for Character Set.
- 'F' for Form Definition.
- 'O' for Overlay.
- 'S' for Page Segment.
- 'T' for Codepage.
- 'X' for Coded Font.

Other characters are ignored.

Action: Check the specifications on the command line.

ARM3005W Generated resource library 'FILE' is empty

Explanation: The output library is empty, either no resources are specified or no resources can be found. The resource library is not usable.

Action: Check the specifications.

ARM3006W Unexpected CRLF byte(s) encountered (removed)

Explanation: AFP files consist of binary records that are stored consecutively without any further information in between. Some programs (e.g., file transfer) add carriage control and line feed characters between the records. The CRLF characters are not valid and other applications issue errors.

Action: Check the source of your AFP data.

ARM3007W Unexpected Ctrl-Z byte encountered (removed)

Explanation: Working with DOS some text editors add the character Ctrl-Z (0x1A) at the end of a file. The character is invalid in AFP files.

Action: Check the source of your AFP data.

ARM3008W Too many entries, extensions 'EXTENSION' ignored

Explanation: It is not possible to specify any number of file extensions for a resource type on the command line.

Action: Check your specifications on the command line.

ARM3009W Extension 'EXTENSION' too long, truncated to 8 characters

Explanation: File extensions for resource types are truncated to 8 characters. Extensions longer than 8 characters are not valid.

Action: Check your specifications on the command line.

ARM4000I Termination with return code RC

Explanation: If an error occurs during processing with a return code not equal to 0 this message is issued.

Action: Check previous error messages.

ARM4001I Resource library 'FILE' created successfully

Explanation: If processing ends error free this message is issued.

Action: Not necessary.

ARM4002I Resource library 'FILE' created, some resources are missing

Explanation: Processing terminates with some errors. The generated resource library is incomplete.

ARM4003I No resource library was specified

Explanation: No output file is specified, i.e., resources are identified and checked, but not assembled in the output file.

Action: Specify an output file with option -output, if you want to generate a library.

CSV - CSV Comma Separated Value

CSV1000F Error allocating column data

Explanation: An error occurred when allocating the definition of a single column. Processing will be aborted.

Action: Contact LRS technical support staff.

CSV1001F Error reading character from input stream

Explanation: An error occurred during reading of a character from the CSV file. The processing can't be continued.

Action: Double check the input file.

CSV2000E Max number of columns (COUNT) reached, truncated

Explanation: The CSV file defines too many columns, only the given number of columns is supported. The additional columns are ignored.

Action: Remove, if possible, unused columns.

CSV2001E Field truncated to 'CHARS' characters

Explanation: A field can only contain the given number of characters, the additional characters are truncated.

Action: Check the input file.

CSV2002E 'CHAR' is not a valid character for a column name

Explanation: Only the following characters may be used in a column name:

- The special character '$'.
- The special character '_'.
- The digits '0' up to '9'.
- The lower Latin letters 'a' up to 'z'.
- The upper Latin letters 'A' up to 'Z'.

Action: Correct the column name.

CSV3000W Unexpected empty line ignored

Explanation: Before the line with the column names there may be no empty line. It is therefore ignored.

Action: Check the input file.

CSV3001W Unexpected end of line, column 'NAME' truncated

Explanation: A date line has less columns than were specified in the columns definition. The missing columns are treated as empty fields.

Action: Check the input file.

CSV3002W Additional columns after column 'NAME' at end of line ignored

Explanation: A data line has more columns than were specified in the columns definition. The additional columns are ignored.

Action: Check the input file.

EXP - Formula Language

EXP1000F Read error

Explanation: An error occurred while processing of an expression of the formula language.

Action: Contact LRS technical support staff.

EXP1001F Error allocation internal expression buffer

Explanation: An error occurred during creation of the internal buffer for expression processing. The formula syntax can't be used.

Action: Contact LRS technical support staff.

EXP1500S Operator expected

Explanation: An operator like '+' or '==' was expected.

Action: Double check your expression.

EXP1501S Unexpected end of data, string termination is missing

Explanation: A string constant has to be terminated with the character ' (apostrophe) or " (quotation) with which it has started.

Action: Double check your string constant.

EXP1502S Unexpected end of data

Explanation: The expression terminates prematurely or wrong.

Action: Double check your expression.

EXP1503S Unexpected end of parameter list

Explanation: The expression terminates in the parameter list of a function.

Action: Double check your function call.

EXP1504S Missing right parenthesis

Explanation: The number of left braces '(' exceeded that of the right ')'.

Action: Double check your parenthesis.

EXP1505S Value expected at this point

Explanation: After an operator like '+' or ==' or a sign like '-' a value must be given.

Action: Double check your expression.

EXP1506S Type mismatch between 'TYPE' and 'TYPE'

Explanation: The values with the given types can't be connected. For example a number can't be subtracted off a string.

Action: Double check the types of the values.

EXP1507S Variables not supported

Explanation: The formula parser context doesn't allow the usage of variables: only constant values are accepted.

Action: Double check your expression.

EXP1508S Variable 'VARIABLE' not found

Explanation: The given variable doesn't exist.

Action: Double check the variable name.

EXP1509S Division by zero

Explanation: The operators '/' (division) and '%' (modulus) must not contain a 0 as second operand, because division by 0 is forbidden.

Action: Double check your operation.

EXP1510S Array index must be of type 'int' not 'TYPE'

Explanation: The index of an array variable in [...] has to be an integer.

Action: Double check your index expression.

EXP1511S Name expected after '.'

Explanation: A name must follow after a colon in a variable expression.

Action: Double check your variable expression.

EXP1512S Name must not be an empty string

Explanation: A single variable name must not be empty.

Action: Double check your variable expression.

EXP1513S Unexpected character in name expression

Explanation: Only the following characters may be used in a name:

- The special character '$';
- The special character '_';
- The digits '0' up to '9';
- The lower latin letters 'a' up to 'z';
- The upper latin letters 'A' up to 'Z';

Furthermore variables can be indexed by [...], wherein an arbitrary expression can occur between the brackets with a result of a non-negative integer (therefore including 0). A colon in-between two names defines a structure element.

Action: Double check your name expression.

EXP1514S Illegal name start character

Explanation: At the beginning of a name only following characters are allowed:

- The special character '$';
- The special character '_';
- The lower latin letters 'a' up to 'z';
- The upper latin letters 'A' up to 'Z';

Action: Double check your name expression.

EXP1515S Function 'FUNCTION' is not defined

Explanation: The given function doesn't exist.

Action: Double check your function call.

EXP1516S Function 'FUNCTION' needs at least COUNT parameter(s)

Explanation: The given function needs at least the announced number of parameters (or more). Too few parameters were given.

Action: Double check your function call.

EXP1517S Function 'FUNCTION' needs COUNT parameter(s)

Explanation: The function given needs exactly the announced number of parameters. Too few (or too many) parameters were given.

Action: Double check your function call.

EXP1518S Function 'FUNCTION' returned RC

Explanation: During function invocation an internal error occurred.

Action: Contact LRS technical support staff.

EXP1519S Illegal 'TYPE' parameter #NUMBER in function 'FUNCTION', 'TYPE' expected

Explanation: The given parameter (counting starts with 1) has a different type than the function expects.

Action: Double check your function call.

EXP2000E More than one decimal point in floating point constant

Explanation: No more than one decimal dot may occur in a floating point constant.

Action: Double check your expression.

EXP2001E Unexpected decimal point in 'octal' or 'hexadecimal' number

Explanation: If a numeric constant starts with '0x' or '0', only an integer may be given, i.e., a decimal dot isn't allowed.

Action: Double check your expression.

EXP2002E Illegal digit 'CHAR' in octal number

Explanation: An octal number starts with '0' and may contain only the digits '0' up to '7'. The digits '8' and '9' aren't allowed in octal numbers.

Action: Double check your expression.

EXP2003E The '+' or '-' sign may only be applied to numeric expressions

Explanation: Only numerical expressions may carry a sign. Strings or logical values must not have sign prefixes.

Action: Double check your expression.

EXP2004E The '!' sign may only be applied to boolean expressions

Explanation: The logical negation, which inverts the logical value, can only be used for logical (boolean) expressions.

Action: Double check your expression.

EXP2005E The '~' sign may only be applied to integer expressions

Explanation: The bit wise not sign, which is used to get the complement, can only be used for integer values.

Action: Double check your expression.

EXP2006E Illegal cast from 'TYPE' to 'TYPE'

Explanation: The type of an expression can be converted (casted) by prefixing with a type in braces, for example (int). But this conversion isn't applicable for all types. For example an integer expression can't be casted to a logical expression.

Action: Double check your expression.

EXP2007E Unexpected end of data after '\' escape sequence introducer

Explanation: In a string constant, which is enclosed by "...", a special character must be given after a '\', but the expression terminates unexpected.

Action: Double check your string constant.

EXP2008E Illegal '\' escape sequence character 'CHAR'

Explanation: In a "..." enclosed string constant a '\' must start one of the following sequences:

- **\b:** backspace - character left, Unicode character U+0008
- **\f:** form feed - page advance, Unicode character U+000C
- **\n:** new line - next line, Unicode character U+000A
- **\r:** carriage return - line back, Unicode character U+000D
- **\t:** tab - tabulator, Unicode character U+0009
- **\\:** backslash itself, Unicode character U+005C

Furthermore also other characters may be encoded with their value:

- **\xnn:** hexadecimal number with up to two digits 0-9 or a-f or A-F for n
- **\nnn:** octal number with up to three digits 0-7 for n
- **\unnnn:** Unicode character with up to four digits 0-9 or a a-f or A-F for n.

Action: Double check your string constant.

EXP2009E Unmatched quotes in format string

Explanation: Text constants in a format string may be started with ' (apostrophe) or " (quotation), but they have to be closed as well with it.

Action: Double check your format string.

EXP2010E Illegal type 'TYPE' for time value, time formatting not possible

Explanation: Time formatting can only be applied to values of type int. Other value types aren't allowed. Result is an empty string.

Action: Double check your expression for time formatting.

EXP2011E Illegal time value, time formatting not possible

Explanation: Time formatting needs a valid time value. Is the time value invalid (for example negative), an empty string is returned.

Action: Double check your expression for time formatting.

EXP2012E Illegal type 'TYPE' for date value, date formatting not possible

Explanation: Date formatting is only applicable on values of type int. Other value types are not accepted. As result an empty string is returned.

Action: Double check your expression for date formatting.

EXP2013E Illegal date value, date formatting not possible

Explanation: Date formatting requires a valid date value. An invalid date value (like 0) results in an empty string.

Action: Double check your expression for date formatting.

EXP2014E Illegal character in string format, formatting not possible

Explanation: String formatting supports a number of so-called tokens (for example FirstCap) as format string. These tokens may contain only letters.

Action: Double check your string formatting expression.

EXP2015E Illegal token 'TOKEN' in string format, formatting not possible

Explanation: String formatting supports a number of so-called tokens (for example FirstCap) as format string. The given token is not supported.

Action: Double check your string formatting expression.

EXP2016E Illegal type 'TYPE' for string format 'TOKEN', formatting not possible

Explanation: During string formatting strings and integer values can be formatted. The value to be formatted must fit to the format, for example a conversion from "abc" into a roman number is impossible.

Action: Double check your string formatting expression.

EXP2017E Illegal type 'TYPE' for number format, formatting not possible

Explanation: Number formatting can be applied only to numerical values, i.e., to integers or floating point numbers.

Action: Double check your number formatting expression.

EXP3000W Name is truncated to NUMBER significant characters

Explanation: The variable name is internally stored only with a maximum number of characters. Maximum exceeding characters are ignored.

Action: Use shorter variable names or check, if the number of significant characters is sufficient for discriminating variables.

EXP3001W Unexpected 'TYPE' parameter #NUMBER in function 'FUNCTION' ignored

Explanation: More parameters then required were given for this function. The additional parameters are ignored.

Action: Double check your function call.

RDI - SAP RDI Parser

RDI1000F Error allocation internal RDI instance data

Explanation: An error occurred during creation of the internal instance for reading RDI files.

Action: Contact LRS technical support staff.

RDI1001F Read error

Explanation: An error occurred while reading a character from the RDI file.

Action: Contact LRS technical support staff.

RDI1002F The file is not a valid RDI file

Explanation: The file is not a valid SAP RDI file. The expected RDI header record is missing.

Action: Check the given RDI file.

RDI2000E The RDI does not contain a valid data record

Explanation: The RDI file does not contain a usable data record and therefore does not produce any output.

Action: Check the given RDI file.

RDI2001E Line too long, truncated

Explanation: The given line of the RDI file is too long and was truncated at the given line position.

Action: Check your RDI input file.

RDI3000W Line starting with 'CHAR' is not expected at this point, ignored

Explanation: The line with the given start character is not expected at this point of a RDI file and is therefore ignored.

Action: Check the given RDI file.

RDI3001E Unexpected end of RDI data, '%%LINES-END' expected

Explanation: A logical RDI record is embedded between a '%%LINES-START' and a '%%LINES-END' line. The '%%LINES-END' line is missing.

Action: Check the given RDI file.

RDI3002E Only lines starting with 'D' expected in RDI data record, 'CHAR line ignored

Explanation: Within a logical RDI record only lines starting with 'D' are expected, other lines are ignored.

Action: Check the given RDI file.

RDI3000W The character '-' in a RDI name is replaced by '_'

Explanation: SAP RDI allows the minus sign (-) within field names. This character is an operator in the Java Script base transformation formula language and can therefore not be used within variable names. To resolve this problem minus characters in RDI field names are replaced by the underscore character (_). This has to be taken into account when referencing such a field name. This message is issued only once.

Action: Change, if required, the field names in the SAP definitions.

XML - Extensible Markup Language Parser

XML1000F Error allocating internal data buffer

Explanation: The internal data area for XML data storage couldn't be created. The processing can't be continued.

Action: Contact LRS technical support staff.

XML1001F Error allocating attribute data

Explanation: The internal data area for XML attribute data storage couldn't be created. The processing can't be continued.

Action: Contact LRS technical support staff.

XML1002F Error allocating entity data

Explanation: The internal data area for XML entity storage couldn't be created. The processing can't be continued.

Action: Contact LRS technical support staff.

XML1003F Error allocating recording list

Explanation: The list for storing of the XML recording files couldn't be created. The processing can't be continued.

Action: Contact LRS technical support staff.

XML1004F Error allocating recording event data

Explanation: The list for storing of the XML recording events couldn't be created. The processing can't be continued.

Action: Contact LRS technical support staff.

XML1005F Error adding name 'NAME' to internal names buffer

Explanation: An element for internal managing of element and attribute names couldn't be created. The processing can't be continued.

Action: Contact LRS technical support staff.

XML1006F Error creating input stream (RC)

Explanation: During initialization of the XML parser a problem occurred with creation of an extended IO object. The processing can't be continued.

Action: Contact LRS technical support staff.

XML1007F Illegal internal state (event=EVENT, state=STATE)

Explanation: An internal error in the status logic of the XML parser occurred, which aborts the execution.

Action: Contact LRS technical support staff.

XML1008F Error opening input file 'FILE' (RC)

Explanation: The given input file can't be opened. Several reasons are possible:

- The file doesn't exist.
- You have no read access for the drive/file system/directory.
- The requested file is locked by another application.

Action: Double check the filename.

XML1009F Error opening external file 'FILE' (errno=ERRNO)

Explanation: The input file to be inserted can't be opened. Several reasons are possible:

- The file doesn't exist.
- You have no read access for the drive/file system/directory.
- The requested file is locked by another application.

Action: Double check the filename.

XML1010F Error reading character from input stream

Explanation: An error occurred during reading of a character from the XML file. The processing can't be continued.

Action: Double check the input file.

XML1011F Unexpected end of file

Explanation: The file ends unexpected in a XML element, i.e., the file is incomplete. The processing can't be continued.

Action: Double check the end of your file.

XML1012F Unexpected end of string

Explanation: The file ends unexpected in a ' or " enclosed string, i.e., the file is incomplete. The processing can't be continued.

Action: Double check the end of your file.

XML1013F Unexpected end of comment

Explanation: The file ends unexpectedly in a comment to be enclosed in <!-- .. -->, i.e., the file is incomplete. The processing can't be continued.

Action: Double check the end of your file.

XML1014F Unexpected end of CDATA

Explanation: The file ends unexpectedly in a string to be enclosed in <![CDATA[..]]>, i.e., the file is incomplete. The processing can't be continued.

Action: Double check the end of your file.

XML1015F Unexpected end of processing instruction

Explanation: The file ends unexpectedly in a processing instruction to be enclosed in <? .. ?>, i.e., the file is incomplete. The processing can't be continued.

Action: Double check the end of your file.

XML1016F Illegal UTF-16 byte order marker

Explanation: While a XML file is opened, an attempt to determine the codepage is done, to interpret the first line. As character set ASCII and EBCDIC are possible. Aside from this in XML additionally a 16 bit character set (Unicode transfer format 16 bit) can be used, with the two start bytes 0xFF and 0xFE (little-endian) or 0xFE and 0xFF (big-endian). While the first start byte was found, the second isn't the matching one. Therefore the file can't be read.

Action: Double check the character set of the input file.

XML1017F Unexpected UTF-16 big-endian encoding

Explanation: While a XML file is opened, a trial to determine the codepage is done, to interpret the first line. As character set ASCII and EBCDIC are possible. Aside from this in XML additionally a 16 bit character set (Unicode transfer format 16 bit) can be used, either indicated by a byte order mark or with the '<' character as 16 bit character. While the first byte 0x00 was found, the second had not the ASCII code of '<'. Therefore the file can't be read.

Action: Double check the character set of the input file.

XML1018F XML version not supported

Explanation: The first statement of an XML file has to announce the XML version as ... version="1.0" .... The current XML parser version supports only this version 1.0. Other versions are unsupported and the processing is aborted.

Action: Double check your input file.

XML1019F Illegal DTD declaration terminates parsing

Explanation: During reading of the Document Type Definition (DTD) an invalid definition was encountered. A faulty DTD aborts the processing.

Action: Double check your DTD or its subset in your document.

XML2000E Syntax error

Explanation: A general syntax error in a <! ... > sequence has occurred. The sequence is ignored.

Action: Double check your input file.

XML2001E Comment must start with ‘<!--’ (second ‘-’ missing)

Explanation: A comment in XML starts with <!-- and ends with -->. Comments can't be nested. In any case two '--' are required, one '-' isn't correct.

Action: Double check your input file.

XML2002E Character reference must be terminated with ';'

Explanation: A character reference is introduced by '&#' and finished with ';'. The terminal ';' is missing.

Action: Double check your input file.

XML2003E Illegal character in character reference

Explanation: A character reference is given either decimal (for example &#32;) or hexadecimal (for example &#x20;), i.e., only decimal and hexadecimal digits are acceptable. Other characters trigger this message.

Action: Double check your input file.

XML2004E Character reference may not be empty

Explanation: A character reference is introduced with '&#' and finished with ';'. In between only a number is allowed, which defines the character code. If there are no digits, this message is triggered.

Action: Double check your input file.

XML2005E Illegal entity reference

Explanation: A reference to an entity is introduced with '&'. If no '#' follows (character entity), a valid XMLname must follow. A valid name starts with '_', a ':' or a letter, including national special letters. The first character after '&' doesn't comply with this rule, therefore the entity reference is ignored.

Action: Double check the character after the '&'.

XML2006E Entity reference must be terminated with ';'

Explanation: A reference to an entity starts with '&' and ends with ';'. In between must be a valid XML name. The ';' wasn't given though.

Action: Double check your input file.

XML2007E Illegal character in character reference

Explanation: A reference to an entity starts with '&' and ends with ';'. The characters in-between must compose a valid XML name, which (with the exception of the first one) must contain this:

- the character '_';
- the character ':';
- the character '-';
- the character '.';
- a letter (inclusive national specials);
- a digit '0' up to '9';
- some more very seldom used characters.

An invalid character was used and therefore the reference ignored.

Action: Double check your input file.

XML2008E Unknown entity '&ENTITY;'

Explanation: The given general entity (with '&...;') wasn't defined in the DTD. The reference to this entity is ignored.

Action: Double check the entity name.

XML2009E Unknown entity '%ENTITY;'

Explanation: The given parameter entity (with '%...;') wasn't defined in the DTD. The reference to this entity is ignored.

Action: Double check the entity name.

XML2010E Processing instruction must be terminated with '?>'

Explanation: A processing instruction is defined by the sequence <? ... ?>. The end is different from other statements indicated with '?>' (and not with '>'). The missing '?' is accepted and the processing instruction done.

Action: Double check your input file.

XML2011E ']]>' in PCDATA must be coded as ']]&gt;'

Explanation: Text in-between two elements is designated as character data. These data are either interpreted (parsed), then they are labeled as parsed character data (PCDATA) or they are taken without interpretation, then they are just called character data (CDATA). Because the interpretation is usually desired (for example with &...; defined special characters), CDATA must be enclosed in a sequence <[CDATA[...]]>. To avoid confusion of interpreted and not interpreted data while sequence termination, this sequence should be noted in PCDATA always with the character reference &amp; instead of a simple &.

Action: If necessary, change your input file.

XML2012E '--' in comment is not allowed

Explanation: A comment in XML is build by the sequence <!-- ... -->. To avoid confusion with the comment termination the occurrence of two subsequent '-' is forbidden in the comment.

Action: Double check the text in the comment.

XML2013E Illegal target name, processing instruction ignored

Explanation: The target name of a processing instruction has to be a valid XML name. In this name not allowed characters were used and therefore the processing instruction ignored.

Action: Double check your input file.

XML2014E Illegal white space ignored

Explanation: In an XML file not at every place are spaces acceptable. For example no space is allowed between the '<' and an element name. The space is ignored and the processing continued.

Action: Double check your input file.

XML2015E Illegal declaration ignored

Explanation: Certain XML sequences start with '<!' and can trigger the following functions:

- If a name is given, a XML declaration follows;
- If '--' is given, a XML comment follows;
- If '[CDATA[' is given, CDATA characters follow.

All other cases trigger this error message.

Action: Double check your input file.

XML2016E Illegal declaration name ignored

Explanation: After the sequence <! only one of the following keywords (if a name is next) may occur:

- DOCTYPE
- ENTITY
- ELEMENT
- ATTLIST

All other keywords lead to this error.

Action: Double check your input file.

XML2017E CDATA must start with proper '<![CDATA[' sequence

Explanation: The sequence <![CDATA[begins data, which are not interpreted (parsed). This error message is displayed, if the '[' after the sequence <![CDATA is missing.

Action: Double check your input file.

XML2018E Illegal " sequence

Explanation: A 'CDATA' or for Microsoft Word XML files an 'if' or 'endif' must follow up the sequence <![. This error message is displayed, if none of the expected keywords is there.

Action: Double check your input file.

XML2019E Character data outside document instance ignored

Explanation: An XML document starts with the so-called root element; that is the one, which defines the highest level of a document instance. Outside of this element (for example in-between the <?xml ... ?> statement and the first element) characters are ignored, which are no spaces.

Action: Double check your input file.

XML2020E Illegal character in attribute list

Explanation: In an element, which is defined by < ... >, only attributes are allowed after the element name, which start with a valid XML name character. All other characters trigger this error message.

Action: Double check your input file.

XML2021E Illegal name start character in attribute

Explanation: In an element, which is defined by < ... >, the element name must follow and be a valid XML name. If an invalid character is used, this error message is triggered.

Action: Double check your input file.

XML2022E Hexadecimal character entity must have small 'x' rather than capital 'X'

Explanation: A character reference is written either decimal (for example &#32;) or hexadecimal (for example &#x20;). The lower case 'x' leads hexadecimal digits. An upper case 'X' isn't foreseen, but is interpreted than the lower case 'x'.

Action: Substitute upper case 'X' with lower case 'x'.

XML3000W Missing byte order marker for UTF-16

Explanation: During opening of an XML file a trial is made, to determine automatically the codepage, for enabling interpretation of the first line. As character set ASCII and EBCDIC are possible. But furthermore in XML a 16 bit character set (Unicode transfer format 16 bit) can be used, which is lead by a so-called byte order marker (0xFEFF). As first character a 0x00 was found and as second one the ASCII code of the '<' character. Therefore the file is interpreted as UTF-16 (big-endian), despite the byte order marker is missing.

Action: Double check the character set of the input file.

XPP - XML Preprocessor

XPP1000F Error creating XIO handle (RC)

Explanation: During initialization of DOPEPREP a problem occurred while creation of an extended IO object. The processing is aborted.

Action: Contact LRS technical support staff.

XPP1001F Error creating XML handle (RC)

Explanation: During initialization of DOPEPREP a problem occurred while creation of the XML parser object. The processing is aborted.

Action: Contact LRS technical support staff.

XPP1002F Error creating varpool (RC)

Explanation: During initialization of DOPEPREP a problem occurred while creation of the varpool object. The processing is aborted.

Action: Contact LRS technical support staff.

XPP1003F Error opening input file 'INPUTFILE' (RC)

Explanation: The command line invocation DOPEPREP input file can't be opened. Following reasons are possible:

- The file doesn't exist;
- You have no read access to the drive/file system/directory;
- The given file is locked by another application.

Action: Double check your call of DOPEPREP.

XPP1004F Error opening output file 'OUTPUTFILE' (RC)

Explanation: The DOPEPREP invocation command line output file can't be opened. Following reasons are possible:

- The file system/drive/directory doesn't exist.
- The file system/drive is (too) full.
- You have no write access to the drive/file system/directory.
- The given file is write protected.
- The given file is currently locked by another application.

Action: Double check your call of DOPEPREP.

XPP1005F Error processing 'INPUTFILE' (RC)

Explanation: During processing of the input file given by the command line with DOPEPREP an error occurred. A more precise problem description was displayed in a previous error message.

Action: Double check your command line and the input file.

XPP1006F Input and/or output file missing

Explanation: A call of DOPEPREP requires two filenames (input and output file).

Action: Double check the DOPEPREP call.

XPP1007F Variable definition 'DEFINITION' expects '='

Explanation: When calling DOPEPREP variables after the option /variable must be written as name=value. The equals character (=) wasn't given or went unrecognized.

Action: Double check the variable declaration. If there are spaces in the expression name=value, the expression has to be set in quotations ("). Aside from this variable names may contain only Latin letters, numbers and the special characters '$' and '_'. The first character must not be a digit.

XPP1008F Error assigning variable 'VARNAME' (RC)

Explanation: During setting a variable in the command line containing DOPEPREP an error occurred. If no previous error message describes the problem in more detail, the following issues may be present:

- An equally named variable already exists with another type.
- The variable name may only contain Latin letters, digits and the special characters '$' and '_'. The first character must not be a digit.

Action: Double check your variable definitions (including the previous ones).

XPP1100F Too many nested levels, element <ELEMENT> ignored

Explanation: The XML-Preprocessor can't process the nesting depth of the XML elements any more. The XML document is too complex.

Action: Simplify your XML file or call LRS technical support.

XPP2000E Unexpected argument 'ARGUMENT' ignored

Explanation: The command line call of DOPEPREP contains too many file names, only two are required.

Action: Double check the DOPEPREP call.

XPP2001E Illegal codepage name 'IANANAME' ignored

Explanation: The call of DOPEPREP contains an invalid character set name within option /codepage. The name has to be a valid IANA character set.

Action: The appendix of the manual lists the supported IANA names. A valid name is for example "ISO-8851-1" or "IBM-850".

XPP2100E Unexpected PCDATA or CDATA in element <ELEMENT> ignored

Explanation: In the given element no text data are allowed. For example in-between <cp:if ...> and a subsequent <cp:then> text isn't acceptable. Spaces and line feeds (whitespace) aren't considered as text in this regard. PCDATA (parsed character data) and CDATA (character data) are notions of the XML vocabulary and correspond to text among XML elements.

Action: Double check your document syntactically.

XPP2101E Illegal 'cp:lang' attribute 'TYPE' in element <ELEMENT> ignored

Explanation: The standard attribute 'cp:lang' requires a valid country/language code. Valid values are for example:

- **de-DE** for German in Germany
- **de-CH** for German in Switzerland
- **en-GB** for English in Great Britain
- **en-US** for English in the United States of America;

Action: Double check the attribute value of 'cp:lang'.

XPP2102E Element 'ELEMENT' not supported in 'cp:' namespace

Explanation: The given element is not supported by the preprocessor language that uses the namespace prefix 'cp:'.

Action: Double check the input file.

XPP2103E Illegal type 'TYPE' in element <ELEMENT>, definition ignored

Explanation: If the attribute type is used in a variable declaration, it must have one of the following values:

- **int** for integer variables.
- **double** for floating point variables.
- **boolean** for logical variables.
- **string** for string/text variables.
- **long-string** for formatted variables.
- **auto** for automatic type recognition.

Action: Double check the value of the type attribute.

XPP2104E Illegal attribute 'ATTRIBUTE' in element <ELEMENT> ignored

Explanation: The allowed attributes are predefined for every element. The given attribute isn't allowed in the given element and is ignored. For example in the element <cp:if ...> the attribute test is allowed, but attribute type is not.

Action: Double check the attributes, which are described for every element in the manual.

XPP2105E Variable 'VARNAME' with type 'int', 'double', 'boolean', 'string' or 'auto' may not be empty

Explanation: In a variable declaration of a certain type the value enclosed between <cp:var> and </cp:var> for example. This value must not be omitted. (If the attribute type isn't present, the value may be omitted. The value is interpreted then as empty string.)

Action: Double check the variable declaration.

XPP2106E Type mismatch for variable 'VARNAME' between 'TYPE' and 'TYPE'

Explanation: The value of the variable doesn't correspond to the announced type. The first displayed type is the one given by attribute type. The second type matches the value encountered. The definition <cp:var type='int'> "Text"</cp:var> would for example produce such an error message, because the expression "Text" is of type string, but the variable is defined with type int.

Action: Double check the expression, which is assigned to the variable.

XPP2107E Error creating variable 'VARNAME' in element <ELEMENT> (RC)

Explanation: During creation of a variable an error occurred. The following reasons are possible:

- Insufficient memory.
- Index out of range.
- Internal processing error.

Action: Double check the variable declaration and if necessary contact LRS technical support.

XPP2108E Character 'CHAR' is not a valid variable name start character for 'VARNAME'

Explanation: The first character of a variable may contain exclusively the following characters:

- Latin upper case letters A - Z;
- Latin lower case letters a - z;
- special character '$';
- special character '_'.

Action: Double check the variable name.

XPP2109E Character 'CHAR' is not a valid variable name character for 'VARNAME'

Explanation: The subsequent characters of a variable may contain exclusively the following characters:

- Digits 0 - 9;
- Latin upper case letters A - Z;
- Latin lower case letters a - z;
- Special character '$';
- Special character '_'.

Action: Double check the variable name.

XPP2110E Attribute 'name' may not be an empty string

Explanation: The attribute name, which determines for example the variable name, must not be empty, i.e., it must not be written as ... name="" ....

Action: Double check the attribute name.

XPP2111E Element <ELEMENT> needs attribute 'name', element ignored

Explanation: Some elements like <cp:var ...> require the attribute name, for example to represent the variable name. Without this attribute the element can't be processed and is ignored.

Action: Double check your input.

XPP2112E Error assigning variable 'VARNAME' (RC)

Explanation: An error occurred during a variable assignment. The following reasons are possible:

- Insufficient memory.
- Internal processing error.

Action: Double check the variable declaration and if necessary contact LRS technical support.

XPP2113E Variable 'VARNAME' already exists as scalar variable, elements ignored

Explanation: A trial is made, to reuse a variable, already defined as scalar (unstructured) one, as a field variable, for example first

```
<cp:var name="Foo">Text</cp:var>
```

and then

```
<cp:array name="Foo">
  <cp:var>different Text</cp:var>
</cp:array>
```

This is impossible.

Action: Double check your variables and if necessary use a different variable name.

XPP2114E Variable 'VARNAME' already exists as array variable, scalar assignment ignored

Explanation: The reuse of an already defined field variable as scalar (unstructured) is tried, for example first

```
<cp:array name="Foo">
  <cp:var>any Text</cp:var>
 </cp:array>
```

and then

```
<cp:var name="Foo">other Text</cp:var>
```

This is impossible.

Action: Double check your variables and if necessary use a different variable name.

XPP2115E Error assigning 'TYPE' value to existing variable 'VARNAME' with type 'TYPE'

Explanation: The reuse of an existing variable with a different type is tried. The first type definition in the message is the new one, the second type definition is the type of the existing variable.

Action: Double check your variables and eventually use a different variable name for another variable.

XPP2116E Attribute 'index' in element <cp:var> must be a positive integer value, ignored

Explanation: The attribute index must be a valid and positive integer (type int). It can be also an expression, which must deliver a positive integer too. Negative indices or even floating point index values must not be used.

Action: Double check your index value or expression.

XPP2117E Illegal attribute 'index' in element <cp:var> ignored

Explanation: The attribute index must be a valid expression. The immediately preceding message displays the error during expression interpretation itself.

Action: Double check your index, even better: the attribute index shouldn't be used any more in a variable declaration with <cp:var>, instead a field variable should be declared always with <cp:array>.

XPP2118E Attribute 'index' within array element <ELEMENT> in array 'VARNAME' ignored

Explanation: The attribute index must not be used for array variables, i.e., the elements <cp:var> or <cp:struct> in <cp:array> must not have an index. The index is automatically incrementally created, starting with 0.

Action: Double check your input and remove the attribute index.

XPP2119E Attribute 'name' within array element <ELEMENT> in array 'VARNAME' ignored

Explanation: The attribute name must not be used for array variables, i.e., the elements <cp:var> or <cp:struct> in <cp:array> have all the same name, which is given in the enclosing <cp:array>.

Action: Double check your input and remove the attribute name.

XPP2120E Element <cp:array> in <cp:array> is not allowed and ignored, use <cp:struct> in-between

Explanation: An array can't be defined immediately below an array (this would introduce multi dimensional arrays). In this regard the following is not allowed:

```
<cp:array name='Row'>
   <cp:array name='Column'>
     <cp:var>Row 0, Column 0<cp:var>
     <cp:var>Row 0, Column 1<cp:var>
   </cp:array>
</cp:array> x
```

But the following is okay:

```
<cp:array name='Row'>
   <cp:struct>
     <cp:array name='Column>
       <cp:var>Row 0, Column 0<cp:var>
       <cp:var>Row 0, Column 1<cp:var>
     </cp:array>
   <cp:struct>
</cp:array>
```

Action: Double check your variable structure.

XPP2121E Element <ELEMENT> has an implicit type, attribute 'type' ignored

Explanation: A structure is a variable of type struct. Therefore no attribute type can be used in element <cp:struct>, <cp:item> (dated) or <cp:record>.

Action: Remove the attribute type.

XPP2122E Only one of the attributes 'while' and 'until' can be specified in <cp:do>, ignored

Explanation: A loop build with <cp:do> can either be a do-while or do-until loop. Therefore only one of both attributes while or until can be used. Because both attributes were used, the entire loop is ignored.

Action: Double check your input.

XPP2123E One of the attributes 'while' or 'until' must be specified in <cp:do>, ignored

Explanation: A loop build with <cp:do> can either be a do-while or do-until loop. Therefore one of the attributes while or until is required. Because no of both attributes was used, the entire loop is ignored.

Action: Double check your input.

XPP2124E Expression in <ELEMENT> must have boolean type, element ignored

Explanation: Certain attributes like the test attribute in element <cp:if> have to return a logical (boolean) value. The value "1 + 2" for example is no logical, but an int type value, while "1 != 2" is a logical value (this evaluates to true). The entire element is ignored.

Action: Double check the expression.

XPP2125E Element <cp:if> needs a 'test' attribute, element ignored

Explanation: The element <cp:if> needs a test attribute, which may contain an expression, which returns a logical value. Because this attribute is missing, the entire if element is ignored.

Action: Add the attribute test.

XPP2126E Element <cp:if> may not be empty, <cp:then> or <cp:else> element expected

Explanation: In-between the element <cp:if> must contain an element <cp:then> or additionally an element <cp:else>. But neither of these sub elements was used.

Action: Double check your input.

XPP2127E Element <cp:then> or <cp:else> may only occur within <cp:if>, element ignored

Explanation: The elements <cp:then> and <cp:else> may occur only as sub elements of <cp:if>. In all other cases these elements are ignored.

Action: Double check your input.

XPP2128E Element <ELEMENT> must not occur more than once within element <cp:if>

Explanation: The elements <cp:then> and <cp:else> may appear each only once in the element <cp:if>. Multiple occurrences are supported though.

Action: Double check your input.

XPP2129E Element <cp:switch> needs a 'test' attribute, element ignored

Explanation: The element <cp:switch> needs a test attribute. Because this attribute is missing, the entire if element is ignored.

Action: Add the attribute test.

XPP2130E Element <cp:switch> needs at least a <cp:case> or <cp:default> child element

Explanation: Within the element <cp:switch> at least one element <cp:case> or <cp:default> must be specified.

Action: Add the missing elements.

XPP2131E Element <cp:case> or <cp:default> may only be used within element <cp:switch>

Explanation: The elements <cp:case> or <cp:default> may only be used within the element <cp:switch>.

Action: Double check your input.

XPP2132E Element <cp:case> needs a 'value' attribute, ignored

Explanation: The elements <cp:case> needs an attribute 'value' to define the case value.

Action: Add the attribute 'value'.

XPP2133E Duplicate value 'VALUE' in element <cp:case> ignored

Explanation: The values in the <cp:case> elements must be unique, the given was value as already been used.

Action: Check your <cp:case> values.

XPP2134E Duplicate element <cp:default> in <cp:switch> ignored

Explanation: The element <cp:default> may only occur once (at the end) within a switch directive.

Action: Check your <cp:default> elements.

XPP2135E Element <cp:case> after <cp:default> ignored

Explanation: If the element <cp:default> is specified within a switch directive, it must be the last element. Therefore <cp:case> elements after the last <cp:default> element are ignored.

Action: Check your <cp:default> elements.

XPP2136E Type mismatch between <cp:switch> attribute 'test' (TYPE) and <cp:case> attribute 'value' (TYPE)

Explanation: The values in the <cp:case> elements must be unique, the given value has already been used.

Action: Check your <cp:case> values.

XPP2137E Element <cp:for> needs the attributes 'variable' (or 'index'), 'begin' and 'end', element ignored

Explanation: The element <cp:for> requires all attributes as follows:

- **variable** as name of the loop iterator,
- **begin** as initial value of the loop iterator variable,
- **end** as final value of the loop iterator variable.

If at least one of these attributes is missing, the entire loop is ignored.

Action: Complete with all attributes.

XPP2138E Attribute 'ATTRIBUTE' in element <cp:for> must be a valid integer value, element ignored

Explanation: The iterator variable in a <cp:for> loop must have type int. Accordingly the values of the attributes begin and end must have type int too.

Action: Double check your input.

XPP2139E Attributes 'index' and 'variable' in element <cp:for> are mutually exclusive, <ELEMENT> ignored

Explanation: The attributes index and variable in a <cp:for> loop can be used alternatively. But only one of these attributes can be used. If both are present, the entire loop is ignored.

Action: Double check your input.

XPP2140E Element <cp:iterate> needs the attributes 'array' (or 'variable') or 'datafile', element ignored

Explanation: The element <cp:iterate> requires either the attribute array (or dated variable) for deciding, which array is to be iterated, or the attribute datafile, to determine the data to build the loop on. If none of both attributes is present, the loop can't be executed and is completely ignored.

Action: Double check your input.

XPP2141E Attribute 'ATTRIBUTE' in element <cp:iterate> must specify an array of structures, element ignored

Explanation: The element <cp:iterate> can iterate only through an array, which has single structures as elements. For a simple array consisting of int values for example <cp:iterate> can't be used (here for example <cp:for> may be applied).

Action: Double check the given array definition.

XPP2142E Attributes 'array' and 'variable' in element <cp:iterate> are mutually exclusive, 'ATTRIBUTE' ignored

Explanation: The attribute variable is a dated convention of the attribute array, i.e., both attributes can be used alternatively. Are both attributes present, the second one is ignored.

Action: Double check your input.

XPP2143E One of the attributes 'name' or 'value' must be specified in <cp:field>, ignored

Explanation: The element <cp:field> substitutes a calculated value (value) or the content of a variable (with name name). Therefore at least one of both attributes must be present.

Action: Double check your input.

XPP2144E Only one of the attributes 'name' and 'value' can be specified in <cp:field>, using 'value'

Explanation: The element <cp:field> substitutes a calculated value (value) or the content of a variable (with name name). With this only one of both attributes can be used. If both are present, the attribute name is ignored.

Action: Double check your input.

XPP2145E Only one 'format...' attribute can be specified in <cp:field>, 'ATTRIBUTE' ignored'

Explanation: In element <cp:field> optionally one of the following format masks can be used:

- **formatdate** for formatting of date values,
- **formattime** for formatting of time values,
- **formatnumber** for formatting of numerical values,
- **formatstring** for formatting of string values.

If more than one of these attributes is used, the other format attributes are ignored.

Action: Double check the format to be used.

XPP2146E The format attribute 'ATTRIBUTE' in <cp:field> is empty and therefore ignored'

Explanation: In element <cp:field> an empty format mask was given. This would result in an empty field content too. Therefore the format mask is ignored.

Action: Double check your format mask.

XPP2147E Element <cp:link> needs one of the attributes 'src' or 'srcexp', element ignored

Explanation: In element <cp:link> the name of the file to include has to be defined either by attribute 'src' (as immediate name) or by the attribute 'srcexp' (as string returning expression). <cp:link> is ignored, because both attributes are missing.

Action: Double check your input.

XPP2148E Only one of the attributes 'src' or 'srcexp' in element <cp:link> expected, 'src' ignored

Explanation: In element <cp:link> the name of the file to include is either given by the attribute 'src' (immediately as name) or by the attribute 'srcexp' (as string returning expression). If both are present, the attribute 'srcexp' is used.

Action: Double check your input.

XPP2149E Attribute 'srcexp' in element <cp:link> must result in a valid string value, element ignored

Explanation: In element <cp:link> the expression of attribute 'srcexp' must result in a valid string value, which is used as filename. The expression is either erroneous or delivers no string value. Therefore the element <cp:link> is ignored.

Action: Double check your input.

XPP2150E Element <cp:index> needs the attributes 'name' and 'value', element ignored

Explanation: The element <cp:index> requires both attributes 'name' and 'value' for assigning an index value. The element is ignored, because one or both attributes are missing,

Action: Double check your input.

XPP2151E Error opening date file 'DATAFILE' (RC)

Explanation: The given data file can't be opened. Possible reasons:

- The file doesn't exist;
- You have no read access to the drive/file system/directory;
- The given file is locked by another application.

Action: Double check the data file.

XPP2152E Illegal value 'DATATYPE' for attribute 'datatype' in element <iterate>

Explanation: The given data type for an external data file is not valid. The following types are supported:

- csv - for "Comma Separated Value" files
- rdi - for SAP "Raw Data Interface" files

Action: Double check your entries.

XPP2153E Illegal language/country definition 'LANG' ignored

Explanation: The definition for language/country is invalid and therefore ignored. You can specify a language with two letters, e.g., "en" for English and a country (separated by '-') can be added, e.g., "en-GB" for British English. Uppercase and lowercase character may be used.

Action: Double check the language/country definition.

XPP2154E Error processing CSV file

Explanation: An error occurred during processing of the entries in the CSV file.

Action: Watch out for the previous messages.

XPP2155E Illegal value 'ENCODING' for attributes 'encoding', assuming 'ISO-8859-1'

Explanation: The value for the attribute encoding must be a valid IANA name like 'ISO-8859-1' or 'IBM850'. The supported encoding values are specified in the appendix of the manual.

Action: Double check your input.

XPP3000W Variable 'VARNAME' truncated to CHARS significant characters

Explanation: Variable names are internally stored only with the given length. Longer variable names are shortened, i.e., the number of significant characters was reduced.

Action: Use shorter variable names or ensure, that the number of significant characters is sufficient for discriminating two variables.

XPP3001W Conversion from 'double' to 'int' for variable 'VARNAME' may loose data

Explanation: Implicit conversion of double to int was performed, e.g., if a variable of type int is used, but the variable value is a floating point number.

Action: Double check your input and if necessary undesired side effects.

XPP3002W Array variable 'VARNAME' defined in element <cp:array> has no elements

Explanation: In-between an <cp:array>...</cp:array> sequence no array elements were defined. Therefore the array variable isn't created, because no empty arrays can be defined.

Action: Double check your input.

XPP3003W Attribute 'count' in element <cp:array> is retired and should not be used any more

Explanation: In earlier versions the number of array elements had to be predefined. This isn't necessary any longer, i.e., arrays grow dynamically.

Action: Remove the attribute count.

XPP3004W Attribute 'format' in element <cp:field> is ignored for 'longstring' variables

Explanation: Variables of type longstring, i.e., variables with a complex XML sequence and all possibilities of formatting, can't be formatted with a format attribute like stringformat.

Action: Double check your input.

XPP3005W Only <cp:else> without <cp:then> encountered within element <cp:if>

Explanation: In-between an <cp:if> element an <cp:else>, but no <cp:then> branch was found.

Action: Double check your logic.

XPP3006W Attribute 'ATTRIBUTE' in conjunction with attribute 'array' is ignored for element <cp:iterate>

Explanation: The given attribute is ignored if 'array' is used as iterator.

Action: Remove unused attributes.

XPP4000I File 'INPUTFILE' successfully processed to 'OUTPUTFILE'

Explanation: The processing of program DOPEPREP was finished successfully without (fatal) errors.

Action: No **Action:** required.

XPP4001I Termination with return code RC

Explanation: The processing of program DOPEPREP was finished with errors. This last message displays the return code of the entire program to the invoking process.

Action: The error causes were revealed in previous messages. Eventually those have to be scrutinized further.

Appendix A
Code Page Names (IANA)

The following table describes the IANA names (Internet Assigned Numbers Authority) of the code pages.

Code Page Names

IANA Name	Description
CP1003	IBM EBCDIC Code Page 1003
CP1004	IBM EBCDIC Code Page 1004
CP1252	ASCII Code Page Windows Latin1
CP2055	IBM EBCDIC Code Page 2055
CP2056	IBM EBCDIC Code Page 2056
CP2063	IBM EBCDIC Code Page 2063
CP2065	IBM EBCDIC Code Page 2065
CP2066	IBM EBCDIC Code Page 2066
CP2067	IBM EBCDIC Code Page 2067
CP2079	IBM EBCDIC Code Page 2079
CP2088	IBM EBCDIC Code Page 2088
CP273	IBM EBCDIC Code Page 273
CP293	IBM EBCDIC Code Page 293
CP361	IBM EBCDIC Code Page 361
CP363	IBM EBCDIC Code Page 363
CP37	IBM EBCDIC Code Page 37
CP382	IBM EBCDIC Code Page 382
CP395	IBM EBCDIC Code Page 395
CP437	ASCII Code Page Latin US
CP500	IBM EBCDIC Code Page Latin US
CP829	IBM EBCDIC Code Page 829
CP850	ASCII Code Page Latin1850
CP852	ASCII Code Page Latin2
CP855	ASCII Code Page Cyrillic
CP857	ASCII Code Page Latin5
CP858	ASCII Code Page Latin1 with Euro Symbol
CP870	IBM EBCDIC Code Page 870
ISO-8859-9	ISO Latin9 (Turkish)
ISO-8895-1	ISO Latin1 (West European)
MacRoman	ASCII Code Page Mac Roman
UTF-8	Special UTF8-Encoding

Table A-1: Code Page Names

Appendix B
Page Formats

The following keywords for page formats are used in attributes of the filter pro- files:

Keyword	Width	Height
4A0	168,2 cm	237,8 cm
2A0	118,9 cm	168,2 cm
A0	84,1 cm	118,9 cm
A1	59,5 cm	84,1 cm
A2	42,0 cm	59,4 cm
A3	29,7 cm	42,0 cm
A4	21,0 cm	29,7 cm
A5	14,8 cm	21,0 cm
A6	10,5 cm	14,8 cm
A7	7,4 cm	10,5 cm
A8	5,2 cm	7,4 cm
A9	3,7 cm	5,2 cm
A10	2,6 cm	3,7 cm

Table B-1: Paper Size - DIN Formats A-Series (ISO 216)

Keyword	Width	Height
B0	100,0 cm	141,4 cm
B1	70,7 cm	100,0 cm
B2	50,0 cm	70,7 cm
B3	35,3 cm	50,0 cm
B4	25,0 cm	35,3 cm
B5	17,6 cm	25,0 cm
B6	12,5 cm	17,6 cm
B7	8,8 cm	12,5 cm
B8	6,2 cm	8,8 cm
B9	4,4 cm	6,2 cm
B10	3,1 cm	4,4 cm

Table B-2: Paper Size - DIN Formats B-Series (ISO 216)

Keyword	Width	Height
C0	91,7 cm	129,7 cm
C1	64,8 cm	91,7 cm
C2	45,8 cm	64,8 cm
C3	32,4 cm	45,8 cm
C4	22,9 cm	32,4 cm
C5	16,2 cm	22,9 cm
C6	11,4 cm	16,2 cm
C65	11,4 cm	22,9 cm
C7	8,1 cm	11,4 cm
C8	5,7 cm	8,1 cm
C9	4,0 cm	5,7 cm
C10	2,8 cm	4,0 cm
DL	11,0 cm	22,0 cm

Table B-3: Paper Size - DIN Formats C-Series (ISO 216)

Keyword	Width	Height
RA0	86,0 cm	122,0 cm
RA1	61,0 cm	86,0 cm
RA2	43,0 cm	61,0 cm
RA3	30,5 cm	43,0 cm
RA4	21,5 cm	30,5 cm

Table B-4: Paper Size - Raw Format A

Keyword	Width	Height
SRA0	90,0 cm	128,0 cm
SRA1	64,0 cm	90,0 cm
SRA2	45,0 cm	64,0 cm
SRA3	32,0 cm	45,0 cm
SRA4	22,5 cm	32,0 cm

Table B-5: Paper Size - Supplementary Raw Format A

Keyword	Width	Height
ID-1	8,560 cm	5,398 cm
ID-2	10,500 cm	7,400 cm
ID-3	12,500 cm	8,800 cm

Table B-6: Paper Size - Identification Cards

Keyword	Width	Height
LETTER	8,500 inch	11,000 inch
LEGAL	8,500 inch	14,000 inch
CSHEET	17,000 inch	22,000 inch
DSHEET	22,000 inch	34,000 inch
ESHEET	34,000 inch	44,000 inch
TABLOID	11,000 inch	17,000 inch
LEDGER	17,000 inch	11,000 inch
STATEMENT	5,500 inch	8,500 inch
EXECUTIVE	7,250 inch	10,500 inch
FOLIO	8,500 inch	13,000 inch
QUARTO	21,500 inch	27,500 inch
10X14	10,000 inch	14,000 inch
11X17	11,000 inch	17,000 inch
NOTE	8,500 inch	11,000 inch
ENV9	3,085 inch	8,075 inch
ENV10	4,125 inch	9,500 inch
ENV11	4,500 inch	10,375 inch
ENV12	4,750 inch	11,000 inch
ENV14	5,000 inch	11,500 inch
MONARCH	3,875 inch	7,500 inch
PERSONAL	3,625 inch	6,500 inch
FANFOLDUS	14,875 inch	11,000 inch
FANFOLDSTDGERMAN	8,500 inch	12,000 inch
FANFOLDLGLGERMAN	8,500 inch	13,000 inch

Table B-7: Paper Size - North America

Keyword	Width	Height
IMPERIALFOLIO	22,000 inch	15,000 inch
IMPERIALQUARTO	15,000 inch	11,000 inch
IMPERIALOCTAVO	11,000 inch	7,500 inch
ROYALFOLIO	20,000 inch	12,500 inch
ROYALQUARTO	12,500 inch	10,000 inch
ROYALOCTAVO	10,000 inch	6,250 inch
CROWNFOLIO	15,000 inch	10,000 inch
CROWNQUARTO	10,000 inch	7,500 inch
CROWNOCTAVO	7,500 inch	5,000 inch
FOOLSCAPFOLIO	13,500 inch	8,500 inch
FOOLSCAPQUARTO	8,500 inch	6,750 inch
FOOLSCAPOCTAVO	6,750 inch	4,250 inch
MEDIUMQUARTO	11,500 inch	9,000 inch
DEMYQUARTO	11,250 inch	8,750 inch
DEMYOCTAVO	8,750 inch	5,625 inch

Table B-8: Paper Size - Imperial

Keyword	Width	Height
CHOU1	14,2 cm	33,2 cm
CHOU2	11,9 cm	27,7 cm
CHOU3	12,0 cm	23,5 cm
CHOU31	10,5 cm	23,5 cm
CHOU30	9,2 cm	23,5 cm
CHOU40	9,0 cm	22,5 cm
CHOU4	9,0 cm	20,5 cm
KAKUA3	32,0 cm	44,0 cm
KAKU0	28,7 cm	38,2 cm
KAKU1	27,0 cm	38,2 cm
KAKU2	24,0 cm	33,2 cm
KAKU3	21,6 cm	27,7 cm
KAKU4	19,7 cm	26,7 cm
KAKU5	19,0 cm	24,0 cm
KAKU6	16,2 cm	22,9 cm
KAKU7	14,2 cm	20,5 cm
KAKU8	11,9 cm	19,7 cm
FURUSU10	23,5 cm	12,0 cm
YOU0	19,7 cm	13,6 cm
YOU1	17,3 cm	11,8 cm
YOU2	16,2 cm	11,4 cm
YOU3	14,8 cm	9,8 cm
YOU4	23,5 cm	10,5 cm
YOU5	21,7 cm	9,5 cm
YOU6	19,0 cm	9,8 cm
YOU7	16,5 cm	9,2 cm

Table B-9: Paper Size - Japanese

Appendix C
SAPGOF Special Characters

The following tables list the SAPGOF special characters that can be specified in the MFFGOF profile. For additional information, see "MFFAFP - User Defined Encoding" on page 3.5.

SAP Number	Unicode Character	Replacement Character	Character Name
35	U+0023	#35	Number Sign
64	U+0040	#64	Commercial AT
92	U+005C	#92	Reverse Solidus
94	U+005E	#94	Circumflex Accent
126	U+007E	#126	Tilde
160	U+00AD	#160	Soft Hyphen
161	U+00A1	#161	Inverted Exclamation Mark
162	U+00A2	#162	Cent Sign
163	U+00A3	#163	Pound Sign
165	U+00A5	#165	Yen Sign
167	U+00A7	#167	Section Sign
168	U+00A4	#168	Currency Sign
171	U+00AB	#171	Left-Pointing Double Angle Quotation Mark
180	U+00B7	#180	Middle Dot
182	U+00A7	#182	Section Sign
187	U+00BB	#187	Right-Pointing Double Angle Quotation Mark
191	U+00BF	#191	Inverted Question Mark
194	U+00B4	#194	Acute Accent
197	U+00AF	#197	Macron
200	U+00A8	#200	Diaeresis
202	U+00B0	#202	Degree Sign
203	U+00B8	#203	Cedilla
225	U+00C6	#225	Latin Capital Letter AE
227	U+00AA	#227	Feminine Ordinal Indicator
233	U+00D8	#233	Latin Capital Letter O With Stroke
235	U+00BA	#235	Masculine Ordinal Indicator
241	U+00E6	#241	Latin Small Letter AE

SAP Number	Unicode Character	Replacement Character	Character Name
249	U+00F8	#249	Latin Small Letter O With Stroke
251	U+00DF	#251	Latin Small Letter Sharp S
256	U+00C1	#256	Latin Capital Letter A with Acute
257	U+00C2	#257	Latin Capital Letter A with Circumflex
258	U+00C4	#258	Latin Capital Letter A with Diaeresis
259	U+00C0	#259	Latin Capital Letter A with Grave
260	U+00C5	#260	Latin Capital Letter A with Ring above
261	U+00C3	#261	Latin Capital Letter A with Tilde
262	U+00C7	#262	Latin Capital Letter C with Cedilla
263	U+00C9	#263	Latin Capital Letter E with Acute
264	U+00CA	#264	Latin Capital Letter E with Circumflex
265	U+00CB	#265	Latin Capital Letter E with Diaeresis
266	U+00C8	#266	Latin Capital Letter E with Grave
267	U+00D0	#267	Latin Capital Letter ETH
270	U+00CD	#268	Latin Capital Letter I with I Acute
271	U+00CE	#271	Latin Capital Letter I with Circumflex
272	U+00CF	#272	Latin Capital Letter I with Diaeresis
274	U+00CC	#274	Latin Capital Letter I with Grave
276	U+00D1	#276	Latin Capital Letter N with Tilde
277	U+00D3	#277	Latin Capital Letter O with Acute
278	U+00D4	#278	Latin Capital Letter O with O Circumflex
279	U+00D6	#279	Latin Capital Letter O with Diaeresis
280	U+00D2	#280	Latin Capital Letter O with Grave
281	U+00D5	#281	Latin Capital Letter O with Tilde
284	U+00DE	#284	Latin Capital Letter THORN
285	U+00DA	#285	Latin Capital Letter U with Acute
286	U+00DB	#286	Latin Capital Letter U with Circumflex
287	U+00DC	#287	Latin Capital Letter U with Diaeresis
288	U+00D9	#288	Latin Capital Letter U with Grave
289	U+00DD	#289	Latin Capital Letter Y with Acute
292	U+00E1	#292	Latin Small Letter A with Acute
293	U+00E2	#293	Latin Small Letter A with Circumflex

SAP Number	Unicode Character	Replacement Character	Character Name
294	U+00E4	#294	Latin Small Letter A with Diaeresis
295	U+00E0	#295	Latin Small Letter A with Grave
296	U+00E5	#296	Latin Small Letter A with Ring above
302	U+00E3	#302	Latin Small Letter A with Tilde
303	U+00A6	#303	Broken Bar
304	U+00E7	#304	Latin Small Letter C with Cedilla
306	U+00A9	#306	Copyright Sign
308	U+00B0	#308	Degree Sign
309	U+00F7	#309	Division Sign
311	U+0060	#311	Grave Accent
312	U+005E	#312	Circumflex Accent
313	U+00EB	#313	Latin Small Letter E with Diaeresis
314	U+00B4	#314	Acute Accent
315	U+00F0	#315	Latin Small Letter ETH
319	U+00ED	#319	Latin Small Letter I with Acute
320	U+00EE	#320	Latin Small Letter I with Circumflex
321	U+00EF	#321	Latin Small Letter I with Diaeresis
322	U+00EC	#322	Latin Small Letter I with Grave
329	U+00AC	#329	Not Sign
332	U+03BC	#332	Greek Small Letter MU
333	U+00D7	#333	Multiplication Sign
335	U+00F1	#335	Latin Small Letter N with Tilde
336	U+00F3	#336	Latin Small Letter O with Acute
337	U+00F4	#337	Latin Small Letter O with Circumflex
338	U+00F6	#338	Latin Small Letter O with Diaeresis
339	U+00F2	#339	Latin Small Letter O with Grave
340	U+00BD	#340	Vulgar Fraction One Half
341	U+00BC	#341	Vulgar Fraction One Quarter
342	U+00B9	#342	Superscript One
343	U+00F5	#343	Latin Small Letter o with Tilde
345	U+00B1	#345	Plus-Minus Sign
347	U+00AE	#347	Registered Sign

SAP Number	Unicode Character	Replacement Character	Character Name
354	U+00FE	#354	Latin Small Letter THORN
355	U+00BE	#355	Vulgar Fraction Three Quarters
356	U+00B3	#356	Superscript Three
358	U+00B2	#358	Superscript Two
359	U+00FA	#359	Latin Small Letter U with Acute
360	U+00FB	#360	Latin Small Letter U with Circumflex
361	U+00FC	#361	Latin Small Letter U with Diaeresis
362	U+00F9	#362	Latin Small Letter U with Grave
364	U+00FD	#364	Latin Small Letter Y with Acute
365	U+00FF	#365	Latin Small Letter Y with Diaeresis
398	U+0169	#398	Latin Small Letter U with Tilde
460	U+2500	#460	Box Drawings Light Horizontal
461	U+2502	#461	Box Drawings Light Vertical
462	U+250C	#462	Box Drawings Light Down and Right
463	U+2510	#463	Box Drawings Light Down and Left
464	U+2514	#464	Box Drawings Light Up and Right
465	U+2518	#465	Box Drawings Light Up and Left
466	U+251C	#466	Box Drawings Light Vertical and Right
467	U+2524	#467	Box Drawings Light Vertical and Left
468	U+252C	#468	Box Drawings Light Down and Horizontal
469	U+2534	#469	Box Drawings Light Up and Horizontal
470	U+253C	#470	Box Drawings Light Vertical and Horizontal
475	U+25B6	#475	Black Right-Pointing Triangle, code page 273
728	U+002E	#728	Full Stop
900	U+20AC	#900	Euro Sign

Table C-3: Encoding List with Prefix Character ‘#’

SAP Number	Unicode Character	Replacement Character	Character Name
0	U+250C	G0	Box Drawings Light Down and Right
1	U+2514	G1	Box Drawings Light Up and Right
2	U+2510	G2	Box Drawings Light Down and Left
3	U+2518	G3	Box Drawings Light Up and Left
4	U+2500	G4	Box Drawings Light Horizontal
5	U+2502	G5	Box Drawings Light Vertical
6	U+251C	G6	Box Drawings Light Vertical and Right
7	U+2524	G7	Box Drawings Light Vertical and Left
8	U+2534	G8	Box Drawings Light Up and Horizontal
9	U+252C	G9	Box Drawings Light Down and Horizontal
10	U+253C	G10	Box Drawings Light Vertical and Horizontal

Table C-4: Encoding List with Prefix Character ‘G’

SAP Number	Unicode Character	Replacement Character	Character Name
58	+002A	D58	Asterisk

Table C-5: Encoding List with Prefix Character ‘D’

Appendix D
Barcodes

Barcodes

Barcodes supported by LRS OEM transformation software products are listed in the table.

Barcode Type	Description
Code 128	Code 128
Code 128B	Code 128B
Code 2/5 Industrial	Code 2/5 Industrial
Code 2/5 Interleaved	Code 2/5 Interleaved
Code 2/5 Matrix	Code 2/5 Matrix
Code 39	Code 39
Code 39 extended	Code 39 extended
Code 93	Code 93
Code 93 extended	Code 93 extended
Data Matrix 200	Data Matrix 200
EAN 128	EAN 128
EAN 13	EAN 13
EAN 8	EAN 8
MSI	MSI
MSI 10	MSI (Modulo 10 check digit)
MSI 11	MSI (Modulo 11 MSI (Modulo 11 check digit))
Patchcode I	Patchcode “I”
Patchcode II	Patchcode “II”
Patchcode III	Patchcode “III”
Patchcode IV	Patchcode “IV”
Patchcode T	Patchcode “T”
Patchcode VI	Patchcode “VI”
POSTNET	POSTNET variable length Barcode
POSTNET 11	POSTNET Delivery Point Barcode (C Prime Field)
POSTNET 5	POSTNET 5-Digit Barcode
POSTNET 9	POSTNET ZIP+4 Barcode (C Field)
UPC A	UPC A
UPC E	UPC E

Table D-6: Barcodes

Appendix E
Documentation

The most recent version of this manual can be downloaded from the LRS Web site (www.lrs.com).

As a licensed user of this product, you may print the PDF file on the *Enterprise Output Management Product Documentation* CD for use within your company as allowed by your license.

Appendix F
Glossary

The glossary explains, in alphabetical order, a multitude of terms relevant to printing, data processing as well as some terms used in the products.

A

ACIF - See AFP Conversion and Indexing Facility.

Additive color mixing - This is the process of producing color through the addition of different colors of light. Computer and television screens use thousands of red, green, and blue phosphor dots, which are so small and close together that the human eye cannot see them individually. Instead, the eye sees the colors formed by the mixture of light.

AFP (Advanced Function Presentation) (1) - A set of licensed programs, together with user applications, that use the all-points-addressable concept to print data on a wide variety of printers or to display data on a variety of display devices. AFP includes creating, formatting, archiving, retrieving, viewing, distributing, and printing information. (2) IBM strategic environment for presentation.

ACIF (AFP Conversion and Indexing Facility) - A batch application development utility that creates documents by formatting line data (record format and traditional), XML data, MO:DCA print files, and unformatted ASCII files. ACIF also provides indexing and resource retrieval capabilities that let you view, distribute, archive, and retrieve document files across systems and platforms.

APA (All Points Addressability) **-** The capability to address, reference, and position text, overlays, and images at any defined position or picture element (pel) on the printable area of a page. This capability depends on the ability of the hardware to address and to display each picture element.

API (Application Programming Interface) **-** A set of routines, protocols, and tools for building software applications. A good API makes it easier to develop a program by providing all the building blocks. A programmer puts the blocks together. Most operating environments, such as MS-Windows, provide an API so that programmers can write applications consistent with the operating environment. Although APIs are designed for programmers, they are ultimately good for users because they guarantee that all programs using a common API will have similar interfaces. This makes it easier for users to learn new programs.

Archiving systems - They are generally used for final filing and are used for revision-secure, unchangeable storage of information. Electronic archiving systems are closely based on a similar approach as classic Document Management Systems. Here, individual documents and containers are administered using a database. Archiving systems also have the possibility of administering large quantities of information in jukeboxes.

ASCII (American Standards Code for Information Interchange) - The standard code, using a coded character set consisting of 7-bit coded characters (8-bits including parity check), that is used for information interchange among data processing systems, data communication systems, and associated equipment. The ASCII set consists of control characters and graphic characters.

Attachment - An attachment is a file such as a word processing document, spreadsheet, graphics or sound that has been sent along with an email message. The contents of the attachment does not appear in the message itself, but instead must be opened separately. The Multipurpose Internet Mail Extensions (MIME) email protocol allows other information such as word processing documents, graphics, sound and video to be sent as electronic mail. This is a method for encoding non-textual data in such a way that it can be transmitted from one computer to another.

B

Bar code **-** (1) A code representing characters by sets of parallel bars of varying thickness and separation that are read optically by transverse scanning. (2) An array of parallel rectangular bars and spaces that together represent data elements or characters in a particular symbology. The bars and spaces are arranged in a predetermined pattern following unambiguous rules defined by the symbology.

Batch - (1) An accumulation of data to be processed. (2) A group of records or data processing jobs brought together for processing or transmission. (3) Pertaining to activity involving little or no user action. Contrast with interactive.

Batch environment - (1) An environment in which noninteractive programs are executed. (2) An environment to which batch jobs in command lists are submitted and in which their execution is scheduled, independently of their submitter.

Batch printing - Queuing one or more documents to print in a separate job as a background process. The operator can work on other documents at the same time the background process is running.

Bitmap - A representation, consisting of rows and columns of dots, of a graphics image in computer memory. The value of each dot (whether it is filled in or not) is stored in one or more bits of data. For simple monochrome images, one bit is sufficient to represent each dot, but for colors and shades of gray, each dot requires more than one bit of data. The more bits used to represent a dot, the more colors and shades of gray that can be represented. The density of the dots, known as the resolution, determines how sharply the image is represented. This is often expressed in dots per inch (dpi) or simply by the number of rows and columns, such as 640 by 480. To display a bit-mapped image on a monitor or to print it on a printer, the computer translates the bit map into pixels (for display screens) or ink dots (for printers). Optical scanners and fax machines work by transforming text or pictures on paper into bit maps. Bit-mapped graphics are often referred to as raster graphics.

C

Character increment - Usually it is the distance between the current print position and the next print position. Synonymous with character spacing.

Character set - (1) A finite set of different characters that is complete for a given purpose; for example, the character set in ISO Standard 646, 7-bit Coded Character Set for Information Processing Interchange. (2) An ordered set of unique representations called characters; for example, the 26 letters of the English alphabet, Boolean 0 and 1, the set of symbols in the Morse code, and the 128 ASCII characters. (3) Often a synonym for font character set. (4) All the valid characters for a programming language or for a computer system. (5) A group of characters used for a specific reason; for example, the set of characters a printer can print.

CMYK - CMYK (an acronym for Cyan, Magenta, Yellow, and Black) designates the color model usually employed in printing technology, which uses the basic colors cyan, magenta and yellow. The fourth “color” is black, which is used to ensure a visually satisfying black tone.

Code page - A set of assignments, each of which assigns a code point to a character. Each code page has a unique name or identifier. Within a given code page, a code point is assigned to one character. More than one character set can be assigned code points from the same code page.

COLD (Computer Output to Laser Disk) **-** The COLD procedure is used for the archiving of mass data on digital optical storage units. COLD systems accept several files, extract index data and save the information. Data can then be displayed in its original format with the use of overlays or background layouts.

Concertina fold - A concertina fold is the continuous parallel folding of brochures and similar printed products in the manner of an accordion, where the fold is alternatively made to the front and back.

Copy group - (1) One or more copies of a sheet of paper or form. Each copy can have modifications, such as text suppression, page position, forms flash, overlays, paper-source, and duplex printing. (2) The portion of a form definition that defines a set of modifications (including identifying the overlays and defining page placement) that can be used when printing a page or form. (3) A set of copy subgroups that specify all copies of a sheet. In IPDS architecture, a copy group is specified by a Load Copy Control command. In MO:DCA, a copy group is specified within a medium map. (4) Synonym for medium map.

CSS (Cascading Style Sheets) **-** A feature being added to HTML that gives both Web site developers and users more control over how pages are displayed. With CSS, designers and users can create style sheets that define how different elements, such as headers and links, appear. These style sheets can then be applied to any Web page. The term cascading derives from the fact that multiple style sheets can be applied to the same Web page. CSS was developed by the W3C.

CSV (Comma Separated Values) **-** It is a type of file usually created with a spreadsheet program: each line (or row) consists of one record, and each field of the record is separated by commas or another user defined character.

D

Data mining - A class of database applications that look for hidden patterns in a group of data that can be used to predict future behavior. For example, data mining software can help retail companies find customers with common interests. The term is commonly misused to describe software that presents data in new ways. True data mining software does not just change the presentation, but actually discovers previously unknown relationships among the data. Data mining is popular in the science and mathematical fields but also is utilized increasingly by marketers trying to distill useful consumer data from Web sites.

DCA (Document Content Architecture) **-** A standard, produced by IBM, for word processor files. Two types of DCA file are used, one for documents that can be altered (“revisable-form text” or RFT) and one for those that cannot (“final-form text” or FFT).

DCF (Document Composition Facility) **-** (1) An IBM licensed program that provides a text formatter called SCRIPT/VS. SCRIPT/VS can process files marked up with a unique set of either SCRIPT or GML controls and tags. (2) In printing, an IBM licensed program that provides text formatting for certain printers, including IBM page printers, IBM line printers, PostScript devices, or any printer functionally equivalent to the IBM printers at the data stream level.

DMS (Document Management System) **-** Total system consisting of hardware and software with which all information existing within an organization can be recorded, administered or handled. This information can be available in paper or electronic form. Paper documents are transferred to digital format prior to be recorded. The document that is to be recorded must be indexed and can then be subjected to a targeted search and retrieved later. Furthermore, referring to the contents of a document allows a full text search to be carried out.

Document - The term applies not only for earlier paper documents, but can also contain any other digital type of information that is linked together into a document. A document can consist of one (e.g., an image or a set of data) or several individual objects (e.g., several images, a file with integrated images, text and tables, a mixture of content from several sources).

Distributed output management - (e.g., IBM's Infoprint, Cypress) is middleware that drives the output process and supports the automated creation and delivery of business process and ad hoc documents.

DPI (Dots per Inch) **-** It indicates the resolution of images. The more dots per inch, the higher the resolution. A common resolution for laser printers is 600 dots per inch. This means 600 dots across and 600 dots down, so there are 360,000 dots per square inch.

E

EBCDIC (Extended Binary Coded Decimal Interchange Code) **-** (1) A coded character set consisting of 8-bit coded characters. (2) A code developed for the representation of textual data. A coded character set of 256 8-bit characters.

EDMS (Electronic Document Management System) **-** Any electronic system that manages documents (not data) to realize significant improvements in business work processes. Like most information systems, EDMSs consist of a number of component technologies that support both analog and digital document management.

Encoding scheme - A set of specific definitions that describe the philosophy used to represent character data. The number of bits, the number of bytes, the allowable ranges of bytes, the maximum number of characters, and the meanings assigned to some generic and specific bit patterns, are some examples of specifications to be found in such a definition.

EPS (Encapsulated PostScript) - Encapsulated PostScript. A standard file format for importing and exporting. Postscript language files among applications in a variety of heterogeneous environments. Pure EPS format cannot be presented on the monitor and therefore provides a preview picture in TIF-Format. EPS graphics can be enlarged without loss of quality.

G

Grammage **-** The standard international unit of measurement for paper weight, expressed in grams per square meter, or gsm. Grammages range from 7 to approx. 225 g/m2 for paper, and approx. 150 to 600 gsm for board. The system differs in the U.S., where the term “basis weight” is used, i.e., the weight in pounds of a ream (500 sheets), in the basic size for a grade of paper.

GIF (Graphical Interchange Format) **-** GIF describes a data format for space-saving storage of images and graphics. It works using lossy compression, and allows 256 colors to be displayed from any pallet. The data format was introduced by the CompuServe online service in 1987, which makes it one of the oldest of its type. GIF remains one of the most widely used formats for online publication, and allows animated images to be displayed in a modified form (.animated GIF.).

I

ICR (Intelligent Character Recognition) **-** It is a method for recognizing text in a NCI (non-coded information) document. In addition to OCR methods, other sources of information are taken into consideration during the conversion, which improves the recognition rate over standard OCR technologies.

Image - (1) An electronic representation of a picture produced by means of sensing light, sound, electron radiation, or other emanations coming from the picture or reflected by the picture. An image can also be generated directly by software without reference to an existing picture. (2) An electronic representation of an original document recorded by a scanning device. (3) A likeness or imitation of an object, such as a picture or logo. (4) In printers, a string of picture elements organized in scan lines to represent the contrasting portions of a picture. The image may consist of any data stored as a raster pattern. An image is printed is one side of a sheet of paper. In simplex printing, a synonym for impression. (5) Toned and untoned pels arranged in a pattern. (6) A PostScript file that can contain any combination of images and text.

IOCA (Image Object Content Architecture) - (1) In printers, an IPDS command set to print image data on a page, page segment, or overlay. (2) An architected collection of constructs used to interchange and present images.

J

Java **-** Java is an object-oriented programming language developed by computer manufacturer Sun Microsystems specifically for Internet applications. So that programs written in Java can be executed on every computer, they are not translated into ‘machine language’ for execution, but into a computer independent code. In a second step during execution, it is then converted into a code that the computer in question “understands”. Every computer that is to execute Java programs needs a program module called a ‘Java Virtual Machine’ for this purpose. This module is available for all common operating systems and is included in the standard Internet browsers.

JPEG (Joint Photographic Experts Group) **-** JPEG is a common method, developed by the Joint Photographic Experts Group, for compressing image files in RGB mode. It can reduce the file size by up to 95 percent. It involves a loss of image information, although the degree of compression can be selected such that these losses remain within acceptable limits. JPEG requires no license and is internationally standardized (ISO 10918). It uses the "Discrete Cosine Transformation" (DCT) method, where image sections of 64 pixels each are processed.

JPEG 2000 - A new compression method, known as JPEG 2000, is currently being developed. It uses what is known as "wavelets" and is said to be able to compress images by 20 percent more than JPEG. The image quality also suffers less at high compression rates, as the image is processed as a whole. Moreover, JPEG 2000 is also to support non-lossy compression, as well as other color modes (such as CMYK) and color management.

L

LZW - Short for Lempel-Zif-Welch, a popular data compression technique developed in 1977 by J. Ziv and A Lempel, and later refined by Terry Welch. It is the compression algorithm used in the GIF graphics file format, which is one of the standard graphic formats used by CompuServe and the World Wide Web. The patent for LZW is owned by Unisys, which for many years allowed anyone to use the algorithm for free. Then in 1995, Unisys suddenly decided to charge a license fee.

M

Medium map - (1) An internal object in a form definition that controls the overlays to be used and defines page placement and modifications to the form. Synonymous with copy group. (2) A print control object in a form map that defines resource mappings and controls modifications to a form, page placement on a form, and form copy generation. Synonym for copy group.

Meta data - Definitional data that provides information about or documentation of other data managed within an application or environment. Meta data may include descriptive information about the context, quality and condition, or characteristics of the data. Also commonly referred to as an abstract.

MICR (Magnetic Ink Character Recognition) **-** (1) Character recognition of magnetic ink characters. (2) The identification of characters using magnetic ink. (3) It is the recognition of characters printed with ink that contains particles of a magnetic material. (4) Contrast with optical character recognition.

MO:DCA (Mixed Object Document Content Architecture) **-** An architected, device-independent data stream for interchanging documents.

MO:DCA-P (Mixed Object: Document Content Architecture for Presentation) **-** The MO:DCA-P data stream is the application data stream through which applications can describe pages composed of text, images, and graphics data. MO:DCA-P is device independent; therefore, applications that produce this data stream can be directed to any of the printers supported by the AFP system or to graphical personal computer displays for viewing.

N

NOP (No Operation) **-** A construct whose execution causes a product to proceed to the next instruction to be processed without taking any other action.

O

OCR (Optical Character Recognition) **-** OCR refers to the branch of computer science that involves reading text from paper and translating the images into a form that the computer can manipulate (for example, into ASCII codes). An OCR system enables you to take a book or a magazine article, feed it directly into an electronic computer file, and then edit the file using a word processor. All OCR systems include an optical scanner for reading text, and sophisticated software for analyzing images. Most OCR systems use a combination of hardware (specialized circuit boards) and software to recognize characters, although some inexpensive systems do it entirely through software. Advanced OCR systems can read text in large variety of fonts, but they still have difficulty with handwritten text.

OMR (Optical Mark Reading) **-** OMR is the technology of electronically extracting intended data from marked fields, such as check boxes and fill-in fields, on printed forms. OMR technology scans a printed form and reads predefined positions and records where marks are made on the form. This technology is useful for applications in which large numbers of hand-filled forms need to be processed quickly and with great accuracy, such as surveys, reply cards, questionnaires and ballots. A common OMR application is the use of "bubble sheets" for multiple-choice tests used by schools. The student indicates the answer on the test by filling in the corresponding bubble, and the form is fed through an optical mark reader (also abbreviated as OMR, a device that scans the document and reads the data from the marked fields. The error rate for OMR technology is less than 1%.

OTF (Output Text Format) **-** The format consists of a set of simple commands, which are not linked to a particular output environment. It is therefore suitable for a whole range of output devices such as line printers, laser printers or monitors. Furthermore, it is easy to read and understand. In SAP applications, OTF serves as a basis for device drivers, i.e., programs that control output to a particular output device.

Overlay - (1) A resource object that can contain text, image, graphics, and barcode data. Overlays define their own environment, and are often used as electronic forms. (2) The final representation of such an object on a physical medium. Contrast with page segment.

P

PDS (Partitioned Data Set) **-** A data set in direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data. Contrast with sequential data set. Synonymous with program library.

Page group - A named group of sequential pages. A page group is delimited by a Begin Named Page Group structured field and an End Named Page Group structured field. A page group can contain nested page groups. All pages in the page group inherit the attributes and processing characteristics that are assigned to the page group.

Page segment - (1) In MO:DCA, a resource object that can contain any mixture of bar code objects, graphics objects, and IOCA image objects. A page segment does not contain an active environment group. The environment for a page segment is defined by the active environment group of the including page or overlay. (2) The final representation of such an object on a physical medium. Contrast with overlay.

PCL (Printer Control Language) **-** PCL is the language used to control computer printers. Introduced in the 1980s by computer manufacturers Hewlett-Packard and under constant development ever since, PCL allows application programs to control the functions of different printers in a standardized, efficient manner. PCL commands are embedded in the data flow of the print job. Compared to the Postscript page description language developed by Adobe, PCL is usually faster and requires less memory. However, the functionality of this language is less general in scope, which means that it is not suitable for all output devices.

PCX - Originally developed by ZSOFT for its PC Paintbrush program, PCX is a graphics file format for graphics programs running on PCs. It is supported by most optical scanners, fax programs, and desktop publishing systems. Files in the PCX format end with a ".pcx".

PDF (Portable Document Format) **-** Developed by software manufacturer Adobe Systems Inc., in the USA, this data format is used for exchanging and processing electronically stored, formatted documents with text and images, independently of the hardware and software used. One of the special features is that texts and graphics are stored in vector form, meaning that the resolution of their representation is dependently solely on the output device (monitor, printer). PDF files can generally by recognized by the ".pdf" file name suffix. They can be created using the Adobe Acrobat program. The Acrobat Reader is available free for displaying and printing PDF files.

PDF/A (Archive) **-** In 2002, AIIM International and NPES (The Association for Suppliers of Printing, Publishing, and Converting Technologies) announced a joint effort to develop an international standard that defines the use of the Portable Document Format (PDF) for archiving and preserving documents. The project, currently referred to as PDF/A, will address the growing need to electronically archive documents in a way that will ensure preservation of their contents over an extended period of time, and will further ensure that those documents will be able to be retrieved and rendered with a consistent and predictable result in the future. This need exists in a growing number of international government and industry segments, including legal systems, libraries, newspapers, regulated industries, and others.

Pixel (Picture Element) **-** (1) A pixel is the computer term for an image dot, i.e., the smallest unit of a digitally displayed image. The memory required by an image consisting of pixels is determined by the size of the image, its resolution, i.e., the number of pixels per unit of area, and the number of colors to be displayed. (2) The format for storing image data where, for a given resolution, every pixel in the image is represented by the appropriate data. Image processing programs such as Photoshop use the pixel format, the most common being TIFF (Tagged Image File Format). The pixel format is most suitable for real images, but, depending on the quality of the image, this requires a very large amount of memory.

PNG (Portable Network Graphics) **-** A bit-mapped graphics format similar to GIF. In fact, PNG was approved as a standard by the World Wide Web consortium to replace GIF because GIF uses a patented data compression algorithm called LZW. In contrast, PNG is completely patent- and license-free. The most recent versions of web browsers now support PNG.

PostScript - PostScript is a page description language developed by the software manufacturer Adobe Systems, Inc., which has become a quasi-standard in the digital prepress stage. It describes documents largely independently of the device used, so that for instance the resolution of an image is not defined until the output device. The more recent PostScript 2 offers, among other things, improved colorimetric facilities, since the reference color space is integrated in accordance with the CIE (Commission Internationale de l'Eclairage) standard. The latest version, PostScript 3, also improves the way in which colors and three-dimensional objects are displayed and supports trapping of graphic objects.

PSF (Print Services Facility) **-** (1) The IBM software product that enables the use of the IBM family of page printers. (2) An IBM licensed program that manages and controls the input data stream and output data stream required by supported IBM page printers. PSF combines print data (from the system spool) with other resources and printing controls to produce the printer data stream (including printer commands) for AFP printers. PSF can interface either directly with a user.s application program or indirectly through the Job Entry Subsystems (JES) of MVS. PSF provides device support for Advanced Function Presentation. (3) A sophisticated print subsystem that drives IPDS page printers. PSF is supported on MVS, VM, and VSE, and as a standard part of OS/400. PSF manages printer resources such as fonts, images, and electronic overlays, and provides error recovery for print jobs. The input data streams supported by PSF include 370 line data and AFPDS/MO:DCA-P.

PTOCA (Presentation Text Object Content Architecture) **-** (1) In printers, the IPDS control sequences used to present text information on a page, page segment, or overlay. (2) An architected collection of constructs used to interchange and present presentation text data.

R

Raster image - A raster image is a matrix of row and column data points whose values represent energy being reflected or emitted from the object being viewed. These values, or pixels, can be viewed on a display monitor as a black and white or color image.

Repository - t generically refers to a central place where data is stored and maintained. A repository can be a place where multiple databases or files are located for distribution over a network, or a repository can be a location that is directly accessible to the user without having to travel across a network.

RGB - RGB is the usual additive color model for the primary colors red, green and blue and is used for self-illuminating output devices such as monitors, but also for electronic recording equipment such as scanners and video cameras. There are a number of different variants of RGB. Hewlett-Packard and Microsoft proposed a new, more uniform standard in 1998 in the form of sRGB.

RTF (Rich Text Format) **-** It is a data format for texts that contains not only the text itself, but also information on the font, font size and formatting. The Rich Text Format was specified by Microsoft as a software-independent format for formatted texts.

S

SAPGOF (SAP Generic Output Format) **-** All external programs which process R/3 spool data (print data) will have to be capable of interpreting the format.

Scale-to-Gray - An option to display a black and white image file in an enhanced mode, making it easier to view. A scale-to-gray display uses gray shading to fill in gaps or jumps (known as aliasing) that occur when displaying an image file on a computer screen. Also known as grayscale.

Serif - A serif is a small stroke on the end of the strokes of letters. This design feature is typical of so-called Roman faces, which can be traced back to ancient Rome. Serifs cater to the perception mechanism of the human eye and thus make typefaces easier to read.

SGML (Standard Generalized Markup Language) **-** It is a system for organizing and tagging elements of a document. SGML was developed and standardized by the International Organization for Standards (ISO) in 1986. SGML itself does not specify any particular formatting; rather, it specifies the rules for tagging elements. These tags can then be interpreted to format elements in different ways. SGML is used widely to manage large documents that are subject to frequent revisions and need to be printed in different formats. Because it is a large and complex system, it is not yet widely used on personal computers. However, the growth of Internet, and especially the World Wide Web, is creating renewed interest in SGML because the World Wide Web uses HTML, which is one way of defining and interpreting tags according to SGML rules.

Spooling - It refers to putting jobs in a buffer, a special area in memory or on a disk where a device can access them when it is ready. Spooling is useful because devices access data at different rates. The buffer provides a waiting station where data can rest while the slower device catches up. The most common spooling application is print spooling. In print spooling, documents are loaded into a buffer (usually an area on a disk), and then the printer pulls them off the buffer at its own rate. Because the documents are in a buffer where they can be accessed by the printer, you can perform other operations on the computer while the printing takes place in the background. Spooling also lets you place a number of print jobs on a queue instead of waiting for each one to finish before specifying the next one.

Spot color - It refers to a method of specifying and printing colors in which each color is printed with its own ink. In contrast, process color printing uses four inks (cyan, magenta, yellow, and black) to produce all other colors. Spot color printing is effective when the printed matter contains only one to three different colors, but it becomes prohibitively expensive for more colors. Most desktop publishing and graphics applications allow you to specify spot colors for text and other elements. There is a number of color specification systems for specifying spot colors, but Pantone is the most widely used.

SQL (Structured Query Language) - SQL is a language for requesting information from a database. The original version called SEQUEL (structured English query language) was designed by an IBM research center in 1974 and 1975. SQL was first introduced as a commercial database system in 1979 by Oracle Corporation.

Subtractive color synthesis - Subtractive color synthesis creates a color impression by filtering out individual frequency ranges (i.e., colors) from the overall spectrum of visible light. In the case of color printing, this is done by overprinting the inks - generally the basic colors cyan, magenta and yellow, which can be used to create all color tones of the relevant color space.

SVG (Scalable Vector Graphics) **-** It is a vector graphics file format that enables two-dimensional images to be displayed in XML pages on the Web. Vector images are created through text-based commands formatted to comply with XML specifications. In contrast to JPEG and GIF images on the Web, which are bitmapped and always remain a specified size, SVG images are scalable to the size of the viewing window and will adjust in size and resolution according to the window in which it is displayed. Benefits of SVG include: smaller files size than regular bitmapped graphics such as GIF and JPEG files, resolution independence, so that the image can scale down or up to fit proportionally into any size display on any type of Web device, text labels and descriptions that can be searched by search engines, ability to link to parts of an image and complex animation.

T

TGA (Targa) **-** It was the first 24-bit truecolor format to come to the PC market. There are several varieties of Targa files; the most commonly used are the Targa 16, Targa 24, and Targa 32. The names are derived from the type of hardware used to create them.

TIFF (Tagged Image File Format) **-** It is a commonly used file format for images. It was defined by a computer industry committee in 1986. It is a so-called screen format that contains information on the brightness and hue of every pixel. The TIFF format supports various color systems, from black and white to full-color RGB images. TIFF files can be compressed by a variety of methods.

Tone correction - It is a method of adjusting the tone curve so that the reproduction results on different types of output devices to have gradations similar to the original image.

True color - It refers to any graphics device or software that uses at least 24 bits to represent each dot or pixel. Using 24 bits means that more than 16 million unique colors can be represented. Since humans can only distinguish a few million colors, this is more than enough to accurately represent any color image.

Truetype - TrueType is a font format developed by Apple and licensed to Microsoft. TrueType is the native operating system font format for Windows and MacOS. TrueType contains a hierarchical set of tables and glyph representations. Characters can be hinted on a per character and point size basis yielding excellent quality at screen resolutions. TrueType fonts for Windows and MacOS have few differences, though they can be different enough to prevent cross platform usage. Font foundries provide TrueType fonts for each platform and usually include a license preventing electronic manipulation to achieve cross platform transparency. TrueType is one of the foundations for the OpenType Font Format.

Twip - Acronym for "twentieth of a point"; a twip is a screen-independent measurement for computer display, used by software program developers. There are 1440 twips in one inch (567 twips per centimeter).

Typeface - It is a design for a set of characters. Popular typefaces include Times Roman, Helvetica, and Courier. The typeface represents one aspect of a font. The font also includes such characteristics as size, weight, italics, and so on. There are two general categories of typefaces: serif and sans serif. Sans serif typefaces are composed of simple lines, whereas serif typefaces use small decorative marks to embellish characters and make them easier to read. Helvetica is a sans serif type and Times Roman is a serif type.

U

Under Color Addition - Under Color Addition (UCA) is a variation of achromatic synthesis used for 4c color sets. Equal components of the three chromatic colors cyan, magenta and yellow in the CMYK color system replace part of the neutral color component generated here with black. Under Color Addition is a good means for improving print quality vis-à-vis pure achromatic synthesis.

Unicode - Unicode is a method for coding characters for electronic processing and uses 16-digit binary numbers (16-bit numbers). Unlike ASCII and other codes, which work with 8-bit numbers, Unicode is capable of representing 65,536 different characters. This covers all the characters is every commonly used script in the world. Unicode is already in widespread use today.

V

Vector graphic - A graphic image drawn in shapes and lines, called paths. Images created in Illustrator and Freehand (graphic design software) are vector graphics. They are usually exported to be bitmap images.

X

XML (Extensible Markup Language, derived from SGML) - It is a symbol language, which is used to describe the structure of documents. Externally, XML is similar to HTML (Hypertext Markup Language) used throughout the Internet. The most important difference lies in the fact that the symbols (“tags”) used in XML can be selected with a great degree of freedom, while they are fixed in HTML. This allows special forms of XML to be generated for virtually any type of application. XML is essentially a slimmed-down form of the SGML (Standard Generalized Markup Language) document description language defined in ISO 8879 and was created for transmitting richly structured documents via the World Wide Web. The International World Wide Web Consortium is responsible for standardization of XML.

XSL (eXtensible Style Definition Language) - XSL is a style sheet language that can be used for displaying XML documents. Using XSL is two-step process, the first step being a transformation of the XML document using XSLT, and the second step being the rendering of the result of the transformation, which is done using XSL-FO. While XSL covers the same application area than CSS, it is much more powerful, because the transformation step (using XSLT) can perform arbitrarily complex transformations of the XML document, while CSS is not able to make any structural changes to the XML document.

XSL-FO (Extensible Style Language-Formatting Objects) - XSL-FO is an XML vocabulary for the formatting of documents. Being part of XSL, the normal way is to produce XSL-FO documents by transforming XML documents using XSLT. Even though the principles behind XSL and CSS (the other Style Sheet Language created by W3C) are quite different, it is planned to align the formatting model between XSL-FO and CSS, so that formatting engines can be based on the same code, both languages can be used to achieve the same results, and formatted results will look identical.

Index