Linux Standard Base Printing Specification 3.2

Linux Standard Base Printing Specification 3.2

Contents

List of Tables

Foreword

This is version 3.2 of the LSB-Printing Module Specification.

An implementation of this version of the specification may not claim to be an implementation of LSB-Printing Module unless it has successfully completed the compliance process as defined by the Linux Foundation.

Introduction

The LSB-Printing Module defines the multimedia components that are required to be present on a conforming system.

This document should be used in conjunction with the documents it references. Information referenced in this way is as much a part of this document as is the information explicitly included here.

I Introductory Elements

1 Scope

The LSB-Printing module defines the multimedia components found on an LSB conforming system.

2 Normative References

The specifications listed below are referenced in whole or in part by the LSB-Printing Module Standard. Such references may be normative or informative; a reference to specification shall only be considered normative if it is explicitly cited as such. The LSB-Printing Module may make normative references to a portion of these specifications (that is, to define a specific function or group of functions); in such cases, only the explicitly referenced portion of the specification is to be considered normative.

Table 2-1 Normative References

Name	Title	URL
ISO C (1999)	ISO/IEC 9899: 1999, Programming Languages --C	
PPD Specification	PostScript Printer Description File Format Specification version 4.3	http://partners.adobe.com/public/developer/en/ps/5003.PPD_Spec_v4.3.pdf
PPD Specification Update	Update to PPD Specification Version 4.3	http://partners.adobe.com/public/developer/en/ps/5645.PPD_Update.pdf

3 Requirements

3.1 Relevant Libraries

The libraries listed in Table 3-1 shall be available on a Linux Standard Base system, with the specified runtime names. This list may be supplemented or amended by the architecture-specific specification.

Table 3-1 Standard Library Names

Library	Runtime Name
libcups	libcups.so.2
libcupsimage	libcupsimage.so.2

These libraries will be in an implementation-defined directory which the dynamic linker shall search by default.

4 Definitions

For the purposes of this document, the following definitions, as specified in the *ISO/IEC Directives, Part 2, 2001, 4th Edition*, apply:

can

: be able to; there is a possibility of; it is possible to

cannot

: be unable to; there is no possibility of; it is not possible to

may

: is permitted; is allowed; is permissible

need not

: it is not required that; no...is required

shall

: is to; is required to; it is required that; has to; only...is permitted; it is necessary

shall not

: is not allowed [permitted] [acceptable] [permissible]; is required to be not; is required that...be not; is not to be

should

: it is recommended that; ought to

should not

: it is not recommended that; ought not to

5 Terminology

For the purposes of this document, the following terms apply:

implementation-defined

Describes a value or behavior that is not defined by this document but is selected by an implementor. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence of the value or behavior. An application that relies on such a value or behavior cannot be assured to be portable across conforming implementations. The implementor shall document such a value or behavior so that it can be used correctly by an application.

Shell Script

A file that is read by an interpreter (e.g., awk). The first line of the shell script includes a reference to its interpreter binary.

undefined

Describes the nature of a value or behavior not defined by this document which results from use of an invalid program construct or invalid data input. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

unspecified

Describes the nature of a value or behavior not specified by this document which results from use of a valid program construct or valid data input. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

6 PPD Format Extensions

The Postscript Printer Description (PPD) format is used in a text file to describe device capabilities for a printing device. PPD files shall conform to the format described by PPD Specification and PPD Specification Update. In addition, several extensions to the standard attribute list are recognized, as listed below. The "cupsVersion" attribute is required in a compliant PPD, while the other attributes are optional.

cupsColorProfile

This string attribute specifies an sRGB-based color profile consisting of gamma and density controls and a 3x3 CMY color transform matrix.

The attribute has the following parameter usage:

*cupsColorProfile Resolution/MediaType: "density gamma m00 m01 m02 m10 m11 m12 m20 m21 m22"

The Resolution and MediaType values may be "-" to act as a wildcard. Otherwise, they must match one of the Resolution or MediaType attributes defined in the PPD file.

The density and gamma values define the gamma and density adjustment function such that (in terms of C math):

f(x) = density * pow(x, gamma)

The m00 through m22 values define a 3x3 transformation matrix for the CMY color values. The density function is applied after the CMY transformation:

| m00 m01 m02 | | m10 m11 m12 | | m20 m21 m22 |

cupsFax

This boolean attribute specifies whether the PPD defines a facsimile device. The default is false.

cupsFilter

The attribute has the following parameter usage:

*cupsFilter: "source/type cost program"

This string attribute provides a conversion rule from the given source type to the printer's native format using the filter "program". A source type is specified according to the conventions of the MIME specification, using "type/subtype" nomenclature, and may refer to a standard MIME type or a CUPS-specific MIME type using the prefix "vnd.cups-" in the subtype. If a printer supports the source type directly, the special filter program "-" may be specified. The cost is an arbitrary positive integer, used to calculate the relative impact a print job has on system load.

cupsManualCopies

This boolean attribute notifies the RIP filters that the destination printer does not support copy generation in hardware. The default value is false.

cupsModelNumber

This integer attribute specifies a printer-specific model number. This number can be used by a filter program to adjust the output for a specific model of printer.

cupsVersion

The attribute has the following parameter usage:

*cupsVersion: "major.minor"

This required attribute describes which version of the CUPS PPD file extensions was used. Currently it must be the string "1.0" or "1.1". The strings "1.2" and "1.3" represent newer versions of the CUPS PPD API that are not covered in this version of the specification, and are currently not allowed, although they may be found in non-conforming PPDs which use a newer version of the CUPS PPD specification.

FoomaticIDs

The attribute has the following parameter usage:

*FoomaticIDs printer driver

The parameters correspond to the IDs in the Foomatic database for the printer and driver, respectively.

FoomaticNoPageAccounting

This boolean attribute tells foomatic-rip whether or not to insert accounting information into the PostScript data stream. By default, foomatic-rip will insert this information.

FoomaticRIPCommandLine

The attribute has the following parameter usage:

*FoomaticRIPCommandLine "code"

This attribute defines the command line in the "code" parameter for the renderer that is called by foomatic-rip. The command must take PostScript on standard input and provide the job data stream in the printer's native language on standard output. The command must exit with status 0 if the conversion was successful and exit with another status if an error occurs. The "code" parameter may contain option setting wildcards, as described below under "FoomaticRIPOption".

FoomaticRIPDefault

The attribute has the following parameter usage:

*FoomaticRIPDefaultOptionName value

This attribute sets a default for a Foomatic option. The name of the attribute should contain the name of the option appended to "FoomaticRIPDefault", with the desired default value as the only parameter.

This option is only used to provide numeric options in the PPD, which are not supported by the Adobe spec, via enumerated options, and should not be used except for that purpose.

FoomaticRIPOption

The attribute has the following parameter usage:

*FoomaticRIPOption name: type style spot [order]

This attribute sets options for the command line specified in the "FoomaticRIPCommandLine" attribute. The "name" parameter specifies the option name, the "type" parameter specifies the option type, the "style" parameter specifies one of "CmdLine", "JCL", "PS", or "Composite", and the "spot" parameter specifies a letter, which is prepended with a "%" and used in the "FoomaticRIPCommandLine" attribute to indicate where the option should go in the command line. The optional "order" parameter indicates an order number for one-choice options.

FoomaticRIPOptionAllowedChars

The attribute has the following parameter usage:

*FoomaticRIPOptionAllowedChars name: "code"

This option sets a list of allowed characters in a string option. The "name" parameter identifies the option, while the "code" parameter is a list of allowed characters.

FoomaticRIPOptionAllowedRegExp

The attribute has the following parameter usage:

*FoomaticRIPOptionAllowedRegExp name: "code"

This option causes the option named by "name" to be validated by the Perl-style regular expression in "code".

FoomaticRIPOptionMaxLength

The attribute has the following parameter usage:

*FoomaticRIPOptionMaxLength name: length

For string or password options, this attribute sets a maximum length which can be returned. The "name" parameter identifies the option, and the "length" parameter is the maximum number of characters allowed.

FoomaticRIPOptionPrototype

The attribute has the following parameter usage:

*FoomaticRIPOptionPrototype name: "code"

For string, password, or simulated numeric options, this attribute sets a code prototype to be inserted into the output. This works for options where the FoomaticRIPOption "style" parameter is set to CmdLine, JCL, or PS. The value of the option can be represented with the string "%s" in the "code" parameter.

FoomaticRIPOptionRange

The attribute has the following parameter usage:

*FoomaticRIPOptionRange name: min max

This attribute adds a minimux and maximum limit to numeric options (that are simulated by Foomatic via emumerated options). The "name" parameter identifies the option, while the "min" and "max" parameters set the minumum and maximum allowed values, respectively, for the option.

FoomaticRIPOptionSetting

The attribute has the following parameter usage:

*FoomaticRIPOptionSetting name=choice: "code"

This attribute adds code to an option, identified by "name", with a FoomaticRIPOption "style" parameter set to Composite. It inserts options for other options that are members of the Composite option "name".

FoomaticRIPPostPipe

The attribute has the following parameter usage:

*FoomaticRIPPostPipe "code"

This attribute defines the command line in the "code" parameter for the job output command used by foomatic-rip in standalone mode. The command should take printer-native data on standard input. The "code" parameter should include the preceding shell pipe symbol ("|").

II LSB Printing Libraries

7 Libraries

7.1 Interfaces for libcups

Table 7-1 defines the library name and shared object name for the libcups library

Table 7-1 libcups Definition

Library:	libcups
SONAME:	libcups.so.2

The behavior of the interfaces in this library is specified by the following specifications:

[LSB] This Specification

7.1.1 CUPS Convenience ABI

7.1.1.1 Interfaces for CUPS Convenience ABI

An LSB conforming implementation shall provide the generic functions for CUPS Convenience ABI specified in Table 7-2, with the full mandatory functionality as described in the referenced underlying specification.

Table 7-2 libcups - CUPS Convenience ABI Function Interfaces

cupsAddDest [LSB]	cupsAddOption [LSB]	cupsCancelJob [LSB]	cupsEncryption [LSB]
cupsFreeDests [LSB]	cupsFreeJobs [LSB]	cupsFreeOptions [LSB]	cupsGetDefault [LSB]
cupsGetDest [LSB]	cupsGetDests [LSB]	cupsGetJobs [LSB]	cupsGetOption [LSB]
cupsGetPPD [LSB]	cupsGetPassword [LSB]	cupsLangEncoding [LSB]	cupsLangFlush [LSB]
cupsLangFree [LSB]	cupsLangGet [LSB]	cupsLastError [LSB]	cupsMarkOptions [LSB]
cupsParseOptions [LSB]	cupsPrintFile [LSB]	cupsPrintFiles [LSB]	cupsServer [LSB]
cupsSetDests [LSB]	cupsSetEncryption [LSB]	cupsSetPasswordCB [LSB]	cupsSetServer [LSB]
cupsSetUser [LSB]	cupsTempFd [LSB]	cupsUser [LSB]	ppdClose [LSB]
ppdCollect [LSB]	ppdConflicts [LSB]	ppdEmit [LSB]	ppdEmitFd [LSB]
ppdEmitJCL [LSB]	ppdErrorString [LSB]	ppdFindAttr [LSB]	ppdFindChoice [LSB]
ppdFindMarkedChoice [LSB]	ppdFindNextAttr [LSB]	ppdFindOption [LSB]	ppdIsMarked [LSB]

ppdLastError [LSB]	ppdMarkDefaults [LSB]	ppdMarkOption [LSB]	ppdOpen [LSB]
ppdOpenFd [LSB]	ppdOpenFile [LSB]	ppdPageLength [LSB]	ppdPageSize [LSB]
ppdPageWidth [LSB]	ppdSetConformance [LSB]		

7.2 Data Definitions for libcups

This section defines global identifiers and their values that are associated with interfaces contained in libcups. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the ISO C (1999) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

7.2.1 cups/cups.h

```
#define _CUPS_CUPS_H_
#define CUPS_VERSION_MAJOR      1
#define CUPS_VERSION_MINOR      1
#define CUPS_VERSION    1.0123
#define CUPS_VERSION_PATCH      23
#define cupsLangDefault()       cupsLangGet(NULL)

typedef enum {
    CUPS_AUTO_ENCODING = -1,
    CUPS_US_ASCII = 0,
    CUPS_ISO8859_1 = 1,
    CUPS_ISO8859_2 = 2,
    CUPS_ISO8859_3 = 3,
    CUPS_ISO8859_4 = 4,
    CUPS_ISO8859_5 = 5,
    CUPS_ISO8859_6 = 6,
    CUPS_ISO8859_7 = 7,
    CUPS_ISO8859_8 = 8,
    CUPS_ISO8859_9 = 9,
    CUPS_ISO8859_10 = 10,
    CUPS_UTF8 = 11,
    CUPS_ISO8859_13 = 12,
    CUPS_ISO8859_14 = 13,
    CUPS_ISO8859_15 = 14,
    CUPS_WINDOWS_874 = 15,
    CUPS_WINDOWS_1250 = 16,
    CUPS_WINDOWS_1251 = 17,
    CUPS_WINDOWS_1252 = 18,
    CUPS_WINDOWS_1253 = 19,
    CUPS_WINDOWS_1254 = 20,
    CUPS_WINDOWS_1255 = 21,
    CUPS_WINDOWS_1256 = 22,
    CUPS_WINDOWS_1257 = 23,
    CUPS_WINDOWS_1258 = 24,
    CUPS_KOI8_R = 25,
    CUPS_KOI8_U = 26
} cups_encoding_t;
typedef struct cups_lang_str {
    struct cups_lang_str *next;
    int used;
    cups_encoding_t encoding;
    char language[16];
    char *messages[506];
} cups_lang_t;
typedef enum {
    HTTP_ENCRYPT_IF_REQUESTED = 0,
    HTTP_ENCRYPT_NEVER = 1,
    HTTP_ENCRYPT_REQUIRED = 2,
    HTTP_ENCRYPT_ALWAYS = 3
} http_encryption_t;
typedef struct {
    char *name;
    char *value;
} cups_option_t;
typedef struct {
    char *name;
    char *instance;
    int is_default;
    int num_options;
    cups_option_t *options;
} cups_dest_t;
typedef enum {
    HTTP_WAITING = 0,
    HTTP_OPTIONS = 1,
    HTTP_GET = 2,
    HTTP_GET_SEND = 3,
    HTTP_HEAD = 4,
    HTTP_POST = 5,
    HTTP_POST_RECV = 6,
    HTTP_POST_SEND = 7,
    HTTP_PUT = 8,
    HTTP_PUT_RECV = 9,
    HTTP_DELETE = 10,
    HTTP_TRACE = 11,
    HTTP_CLOSE = 12,
    HTTP_STATUS = 13
} http_state_t;
typedef enum {
    HTTP_ERROR = -1,
    HTTP_CONTINUE = 100,
    HTTP_SWITCHING_PROTOCOLS = 101,
    HTTP_OK = 200,
    HTTP_CREATED = 201,
    HTTP_ACCEPTED = 202,
    HTTP_NOT_AUTHORITATIVE = 203,
    HTTP_NO_CONTENT = 204,
    HTTP_RESET_CONTENT = 205,
    HTTP_PARTIAL_CONTENT = 206,
    HTTP_MULTIPLE_CHOICES = 300,
    HTTP_MOVED_PERMANENTLY = 301,
    HTTP_MOVED_TEMPORARILY = 302,
    HTTP_SEE_OTHER = 303,
    HTTP_NOT_MODIFIED = 304,
    HTTP_USE_PROXY = 305,
    HTTP_BAD_REQUEST = 400,
    HTTP_UNAUTHORIZED = 401,
    HTTP_PAYMENT_REQUIRED = 402,
    HTTP_FORBIDDEN = 403,
    HTTP_NOT_FOUND = 404,
    HTTP_METHOD_NOT_ALLOWED = 405,
    HTTP_NOT_ACCEPTABLE = 406,
    HTTP_PROXY_AUTHENTICATION = 407,
    HTTP_REQUEST_TIMEOUT = 408,
    HTTP_CONFLICT = 409,
    HTTP_GONE = 410,
    HTTP_LENGTH_REQUIRED = 411,
    HTTP_PRECONDITION = 412,
    HTTP_REQUEST_TOO_LARGE = 413,
    HTTP_URI_TOO_LONG = 414,
    HTTP_UNSUPPORTED_MEDIATYPE = 415,
    HTTP_UPGRADE_REQUIRED = 426,
    HTTP_SERVER_ERROR = 500,
    HTTP_NOT_IMPLEMENTED = 501,
    HTTP_BAD_GATEWAY = 502,
    HTTP_SERVICE_UNAVAILABLE = 503,
    HTTP_GATEWAY_TIMEOUT = 504,
    HTTP_NOT_SUPPORTED = 505
} http_status_t;
typedef enum {
    HTTP_0_9 = 9,
    HTTP_1_0 = 100,
    HTTP_1_1 = 101
} http_version_t;
typedef enum {
    HTTP_KEEPALIVE_OFF = 0,
    HTTP_KEEPALIVE_ON = 1
} http_keepalive_t;
typedef enum {
    HTTP_ENCODE_LENGTH = 0,
    HTTP_ENCODE_CHUNKED = 1
} http_encoding_t;
typedef unsigned int md5_word_t;
typedef unsigned char md5_byte_t;
typedef struct md5_state_s {
    md5_word_t count[2];
    md5_word_t abcd[4];
    md5_byte_t buf[64];
} md5_state_t;
typedef enum {
    IPP_JOB_PENDING = 3,
    IPP_JOB_HELD = 4,
    IPP_JOB_PROCESSING = 5,
    IPP_JOB_STOPPED = 6,
    IPP_JOB_CANCELLED = 7,
    IPP_JOB_ABORTED = 8,
    IPP_JOB_COMPLETED = 9
} ipp_jstate_t;
typedef struct {
    int id;
    char *dest;
    char *title;
    char *user;
    char *format;
    ipp_jstate_t state;
    int size;
    int priority;
    time_t completed_time;
    time_t creation_time;
    time_t processing_time;
} cups_job_t;
typedef enum {
    IPP_OK = 0,
    IPP_OK_SUBST = 1,
    IPP_OK_CONFLICT = 2,
    IPP_OK_IGNORED_SUBSCRIPTIONS = 3,
    IPP_OK_IGNORED_NOTIFICATIONS = 4,
    IPP_OK_TOO_MANY_EVENTS = 5,
    IPP_OK_BUT_CANCEL_SUBSCRIPTION = 6,
    IPP_REDIRECTION_OTHER_SITE = 768,
    IPP_BAD_REQUEST = 1024,
    IPP_FORBIDDEN = 1025,
    IPP_NOT_AUTHENTICATED = 1026,
    IPP_NOT_AUTHORIZED = 1027,
    IPP_NOT_POSSIBLE = 1028,
    IPP_TIMEOUT = 1029,
    IPP_NOT_FOUND = 1030,
    IPP_GONE = 1031,
    IPP_REQUEST_ENTITY = 1032,
    IPP_REQUEST_VALUE = 1033,
    IPP_DOCUMENT_FORMAT = 1034,
    IPP_ATTRIBUTES = 1035,
    IPP_URI_SCHEME = 1036,
    IPP_CHARSET = 1037,
    IPP_CONFLICT = 1038,
    IPP_COMPRESSION_NOT_SUPPORTED = 1039,
    IPP_COMPRESSION_ERROR = 1040,
    IPP_DOCUMENT_FORMAT_ERROR = 1041,
    IPP_DOCUMENT_ACCESS_ERROR = 1042,
    IPP_ATTRIBUTES_NOT_SETTABLE = 1043,
    IPP_IGNORED_ALL_SUBSCRIPTIONS = 1044,
    IPP_TOO_MANY_SUBSCRIPTIONS = 1045,
    IPP_IGNORED_ALL_NOTIFICATIONS = 1046,
    IPP_PRINT_SUPPORT_FILE_NOT_FOUND = 1047,
    IPP_INTERNAL_ERROR = 1280,
    IPP_OPERATION_NOT_SUPPORTED = 1281,
    IPP_SERVICE_UNAVAILABLE = 1282,
    IPP_VERSION_NOT_SUPPORTED = 1283,
    IPP_DEVICE_ERROR = 1284,
    IPP_TEMPORARY_ERROR = 1285,
    IPP_NOT_ACCEPTING = 1286,
    IPP_PRINTER_BUSY = 1287,
    IPP_ERROR_JOB_CANCELLED = 1288,
    IPP_MULTIPLE_JOBS_NOT_SUPPORTED = 1289,
    IPP_PRINTER_IS_DEACTIVATED = 1290
} ipp_status_t;

typedef struct {
    int fd;
    int blocking;
    int error;
    time_t activity;
    http_state_t state;
    http_status_t status;
    http_version_t version;
    http_keepalive_t keep_alive;
    struct sockaddr_in hostaddr;
    char hostname[256];
    char fields[27][256];
    char *data;
    http_encoding_t data_encoding;
    int data_remaining;
    int used;
    char buffer[2048];
    int auth_type;
    md5_state_t md5_state;
    char nonce[256];
    int nonce_count;
    void *tls;
    http_encryption_t encryption;
    fd_set *input_set;
    http_status_t expect;
    char *cookie;
    char authstring[256];
    char userpass[256];
    int digest_tries;
} http_t;
extern void cupsLangFree(cups_lang_t *);
extern void cupsSetEncryption(http_encryption_t);
extern cups_dest_t *cupsGetDest(const char *, const char *, int,
                                cups_dest_t *);
extern int cupsGetJobs(cups_job_t * *, const char *, int, int);
extern http_encryption_t cupsEncryption(void);
extern void cupsFreeJobs(int, cups_job_t *);
extern void cupsFreeOptions(int, cups_option_t *);
extern const char *cupsGetOption(const char *, int, cups_option_t
*);
extern int cupsMarkOptions(ppd_file_t *, int, cups_option_t *);
extern int cupsAddOption(const char *, const char *, int,
                         cups_option_t * *);
extern int cupsGetDests(cups_dest_t * *);
extern void cupsSetServer(const char *);
extern const char *cupsGetPassword(const char *);
extern void cupsSetDests(int, cups_dest_t *);
extern int cupsParseOptions(const char *, int, cups_option_t *
*);
extern void cupsSetPasswordCB(const char *(*digest_tries) (const
char *)
    );
extern void cupsSetUser(const char *);
extern cups_lang_t *cupsLangGet(const char *);
extern void cupsLangFlush(void);
extern int cupsPrintFiles(const char *, int, const char **, const
char *,
                          int, cups_option_t *);
extern int cupsCancelJob(const char *, int);
extern char *cupsLangEncoding(cups_lang_t *);
extern void cupsFreeDests(int, cups_dest_t *);
extern ipp_status_t cupsLastError(void);
extern const char *cupsGetDefault(void);
extern const char *cupsGetPPD(const char *);
extern const char *cupsServer(void);
extern const char *cupsUser(void);
extern int cupsTempFd(char *, int);
extern int cupsPrintFile(const char *, const char *, const char
*, int,
                         cups_option_t *);
extern int cupsAddDest(const char *, const char *, int,
cups_dest_t * *);
```

7.2.2 cups/ppd.h

```
#define _CUPS_PPD_H_
#define PPD_MAX_LINE    256
#define PPD_VERSION     4.3
#define PPD_MAX_NAME    41
#define PPD_MAX_TEXT    81

typedef enum {
    PPD_CS_CMYK = -4,
    PPD_CS_CMY = -3,
    PPD_CS_GRAY = 1,
    PPD_CS_RGB = 3,
    PPD_CS_RGBK = 4,
    PPD_CS_N = 5
} ppd_cs_t;
typedef struct {
    char name[41];
    char *start;
    char *stop;
} ppd_emul_t;
typedef enum {
    PPD_UI_BOOLEAN = 0,
    PPD_UI_PICKONE = 1,
    PPD_UI_PICKMANY = 2
} ppd_ui_t;
typedef enum {
    PPD_ORDER_ANY = 0,
    PPD_ORDER_DOCUMENT = 1,
    PPD_ORDER_EXIT = 2,
    PPD_ORDER_JCL = 3,
    PPD_ORDER_PAGE = 4,
    PPD_ORDER_PROLOG = 5
} ppd_section_t;
typedef struct {
    char marked;
    char choice[41];
    char text[81];
    char *code;
    void *option;
} ppd_choice_t;
typedef struct {
    char conflicted;
    char keyword[41];
    char defchoice[41];
    char text[81];
    ppd_ui_t ui;
    ppd_section_t section;
    float order;
    int num_choices;
    ppd_choice_t *choices;
} ppd_option_t;
typedef struct ppd_group_str {
    char text[40];
    char name[41];
    int num_options;
    ppd_option_t *options;
    int num_subgroups;
    struct ppd_group_str *subgroups;
} ppd_group_t;
typedef struct {
    int marked;
    char name[41];
    float width;
    float length;
    float left;
    float bottom;
    float right;
    float top;
} ppd_size_t;
typedef struct {
    char option1[41];
    char choice1[41];
    char option2[41];
    char choice2[41];
} ppd_const_t;
typedef struct {
    char resolution[41];
    char media_type[41];
    float density;
```

```
        float gamma;
        float matrix[3][3];
    } ppd_profile_t;
    typedef struct {
        char name[41];
        char spec[41];
        char text[81];
        char *value;
    } ppd_attr_t;
    typedef struct {
        int language_level;
        int color_device;
        int variable_sizes;
        int accurate_screens;
        int contone_only;
        int landscape;
        int model_number;
        int manual_copies;
        int throughput;
        ppd_cs_t colorspace;
        char *patches;
        int num_emulations;
        ppd_emul_t *emulations;
        char *jcl_begin;
        char *jcl_ps;
        char *jcl_end;
        char *lang_encoding;
        char *lang_version;
        char *modelname;
        char *ttrasterizer;
        char *manufacturer;
        char *product;
        char *nickname;
        char *shortnickname;
        int num_groups;
        ppd_group_t *groups;
        int num_sizes;
        ppd_size_t *sizes;
        float custom_min[2];
        float custom_max[2];
        float custom_margins[4];
        int num_consts;
        ppd_const_t *consts;
        int num_fonts;
        char **fonts;
        int num_profiles;
        ppd_profile_t *profiles;
        int num_filters;
        char **filters;
        int flip_duplex;
        char *protocols;
        char *pcfilename;
        int num_attrs;
        int cur_attr;
        ppd_attr_t **attrs;
    } ppd_file_t;
    typedef enum {
        PPD_OK = 0,
        PPD_FILE_OPEN_ERROR = 1,
        PPD_NULL_FILE = 2,
        PPD_ALLOC_ERROR = 3,
        PPD_MISSING_PPDADOBE4 = 4,
        PPD_MISSING_VALUE = 5,
        PPD_INTERNAL_ERROR = 6,
        PPD_BAD_OPEN_GROUP = 7,
        PPD_NESTED_OPEN_GROUP = 8,
        PPD_BAD_OPEN_UI = 9,
        PPD_NESTED_OPEN_UI = 10,
        PPD_BAD_ORDER_DEPENDENCY = 11,
        PPD_BAD_UI_CONSTRAINTS = 12,
        PPD_MISSING_ASTERISK = 13,
        PPD_LINE_TOO_LONG = 14,
        PPD_ILLEGAL_CHARACTER = 15,
        PPD_ILLEGAL_MAIN_KEYWORD = 16,
        PPD_ILLEGAL_OPTION_KEYWORD = 17,
        PPD_ILLEGAL_TRANSLATION = 18,
        PPD_ILLEGAL_WHITESPACE = 19
    } ppd_status_t;
    typedef enum {
        PPD_CONFORM_RELAXED = 0,
        PPD_CONFORM_STRICT = 1
    } ppd_conform_t;
    extern float ppdPageLength(ppd_file_t *, const char *);
    extern ppd_status_t ppdLastError(int *);
    extern int ppdEmitFd(ppd_file_t *, int, ppd_section_t);
    extern int ppdMarkOption(ppd_file_t *, const char *, const char
    *);
    extern int ppdEmitJCL(ppd_file_t *, FILE *, int, const char *,
                          const char *);
    extern ppd_choice_t *ppdFindChoice(ppd_option_t *, const char *);
    extern ppd_file_t *ppdOpenFile(const char *);
    extern int ppdEmit(ppd_file_t *, FILE *, ppd_section_t);
    extern int ppdCollect(ppd_file_t *, ppd_section_t, ppd_choice_t *
    **);
    extern ppd_option_t *ppdFindOption(ppd_file_t *, const char *);
    extern void ppdMarkDefaults(ppd_file_t *);
    extern ppd_file_t *ppdOpenFd(int);
    extern ppd_attr_t *ppdFindNextAttr(ppd_file_t *, const char *,
                                       const char *);
    extern const char *ppdErrorString(ppd_status_t);
    extern ppd_attr_t *ppdFindAttr(ppd_file_t *, const char *, const
    char *);
    extern ppd_size_t *ppdPageSize(ppd_file_t *, const char *);
    extern ppd_file_t *ppdOpen(FILE *);
    extern int ppdIsMarked(ppd_file_t *, const char *, const char *);
    extern float ppdPageWidth(ppd_file_t *, const char *);
    extern int ppdConflicts(ppd_file_t *);
    extern ppd_choice_t *ppdFindMarkedChoice(ppd_file_t *, const char
    *);
    extern void ppdClose(ppd_file_t *);
    extern void ppdSetConformance(ppd_conform_t);
```

7.3 Interface Definitions for libcups

The interfaces defined on the following pages are included in libcups and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed in Section 7.1 shall behave as described in the referenced base document.

cupsAddDest

Name

`cupsAddDest` — Add a destination to the list of destinations.

Synopsis

```
#include <cups/cups.h>
int cupsAddDest(const char * name, const char * instance, int
num_dests, cups_dest_t ** dests);
```

Description

Add a destination to the list of destinations.

This function cannot be used to add a new class or printer queue, it only adds a new container of saved options for the named destination or instance.

If the named destination already exists, the destination list is returned unchanged. Adding a new instance of a destination creates a copy of that destination's options.

Use the cupsSaveDests() function to save the updated list of destinations to the user's lpoptions file.

Return Value

New number of destinations

cupsAddOption

Name

`cupsAddOption` — Add an option to an option array.

Synopsis

```
#include <cups/cups.h>
int cupsAddOption(const char * name, const char * value, int
num_options, cups_option_t ** options);
```

Description

Add an option to an option array.

Return Value

Number of options

cupsCancelJob

Name

`cupsCancelJob` — Cancel a print job on the default server.

Synopsis

```
#include <cups/cups.h>
int cupsCancelJob(const char * name, int job);
```

Description

Cancel a print job on the default server.

Use the cupsLastError() and cupsLastErrorString() functions to get the cause of any failure.

Return Value

1 on success, 0 on failure

cupsEncryption

Name

`cupsEncryption` — Get the default encryption settings.

Synopsis

```
#include <cups/cups.h>
http_encryption_t cupsEncryption(void);
```

Description

Get the default encryption settings.

The default encryption setting comes from the CUPS_ENCRYPTION environment variable, then the ~/.cupsrc file, and finally the /etc/cups/client.conf file. If not set, the default is HTTP_ENCRYPT_IF_REQUESTED.

Return Value

Encryption settings

cupsFreeDests

Name

`cupsFreeDests` — Free the memory used by the list of destinations.

Synopsis

```
#include <cups/cups.h>
void cupsFreeDests(int num_dests, cups_dest_t * dests);
```

Description

Free the memory used by the list of destinations.

Return Value

This function does not return a value.

cupsFreeJobs

Name

`cupsFreeJobs` — Free memory used by job data.

Synopsis

```
#include <cups/cups.h>
void cupsFreeJobs(int num_jobs, cups_job_t * jobs);
```

Description

Free memory used by job data.

Return Value

This function does not return a value.

cupsFreeOptions

Name

`cupsFreeOptions` — Free all memory used by options.

Synopsis

```
#include <cups/cups.h>
void cupsFreeOptions(int num_options, cups_option_t * options);
```

Description

Free all memory used by options.

Return Value

This function does not return a value.

cupsGetDefault

Name

`cupsGetDefault` — Get the default printer or class for the default server.

Synopsis

```
#include <cups/cups.h>
const char * cupsGetDefault(void);
```

Description

Get the default printer or class for the default server.

This function returns the default printer or class as defined by the LPDEST or PRINTER environment variables. If these environment variables are not set, the server default destination is returned. Applications should use the cupsGetDests() and cupsGetDest() functions to get the user-defined default printer, as this function does not support the lpoptions-defined default printer.

Return Value

Default printer or NULL

cupsGetDest

Name

`cupsGetDest` — Get the named destination from the list.

Synopsis

```
#include <cups/cups.h>
cups_dest_t * cupsGetDest(const char * name, const char * instance,
int num_dests, cups_dest_t * dests);
```

Description

Get the named destination from the list.

Use the cupsGetDests() or cupsGetDests2() functions to get a list of supported destinations for the current user.

Return Value

Destination pointer or NULL

cupsGetDests

Name

`cupsGetDests` — Get the list of destinations from the default server.

Synopsis

```
#include <cups/cups.h>
int cupsGetDests(cups_dest_t ** dests);
```

Description

Get the list of destinations from the default server.

Starting with CUPS 1.2, the returned list of destinations include the printer-info, printer-is-accepting-jobs, printer-is-shared, printer-make-and-model, printer-state, printer-state-change-time, printer-state-reasons, and printer-type attributes as options.

Use the cupsFreeDests() function to free the destination list and the cupsGetDest() function to find a particular destination.

Return Value

Number of destinations

cupsGetJobs

Name

`cupsGetJobs` — Get the jobs from the default server.

Synopsis

```
#include <cups/cups.h>
int cupsGetJobs(cups_job_t ** jobs, const char * mydest, int myjobs,
int completed);
```

Description

Get the jobs from the default server.

Return Value

Number of jobs

cupsGetOption

Name

`cupsGetOption` — Get an option value.

Synopsis

```
#include <cups/cups.h>
const char * cupsGetOption(const char * name, int num_options,
cups_option_t * options);
```

Description

Get an option value.

Return Value

Option value or NULL

cupsGetPPD

Name

`cupsGetPPD` — Get the PPD file for a printer on the default server.

Synopsis

```
#include <cups/cups.h>
const char * cupsGetPPD(const char * name);
```

Description

Get the PPD file for a printer on the default server.

For classes, cupsGetPPD() returns the PPD file for the first printer in the class.

Return Value

Filename for PPD file

cupsGetPassword

Name

`cupsGetPassword` — Get a password from the user.

Synopsis

```
#include <cups/cups.h>
const char * cupsGetPassword(const char * prompt);
```

Description

Get a password from the user.

Uses the current password callback function. Returns NULL if the user does not provide a password.

Return Value

Password

cupsLangEncoding

Name

`cupsLangEncoding` — Return the character encoding (us-ascii, etc.)

Synopsis

```
#include <cups/cups.h>
const char * cupsLangEncoding(cups_lang_t * lang);
```

Description

Return the character encoding (us-ascii, etc.) for the given language.

Return Value

Character encoding

cupsLangFlush

Name

`cupsLangFlush` — Flush all language data out of the cache.

Synopsis

```
#include <cups/cups.h>
void cupsLangFlush(void);
```

Description

Flush all language data out of the cache.

Return Value

This function does not return a value.

cupsLangFree

Name

`cupsLangFree` — Free language data.

Synopsis

```
#include <cups/cups.h>
void cupsLangFree(cups_lang_t * lang);
```

Description

Free language data.

This does not actually free anything; use cupsLangFlush() for that.

Return Value

This function does not return a value.

cupsLangGet

Name

`cupsLangGet` — Get a language.

Synopsis

```
#include <cups/cups.h>
cups_lang_t * cupsLangGet(const char * language);
```

Description

Get a language.

Return Value

Language data

cupsLastError

Name

`cupsLastError` — Return the last IPP status code.

Synopsis

```
#include <cups/cups.h>
ipp_status_t cupsLastError(void);
```

Description

Return the last IPP status code.

Return Value

IPP status code from last request

cupsMarkOptions

Name

`cupsMarkOptions` — Mark command-line options in a PPD file.

Synopsis

```
#include <cups/cups.h>
int cupsMarkOptions(ppd_file_t * ppd, int num_options, cups_option_t
* options);
```

Description

Mark command-line options in a PPD file.

Return Value

1 if conflicting

cupsParseOptions

Name

`cupsParseOptions` — Parse options from a command-line argument.

Synopsis

```
#include <cups/cups.h>
int cupsParseOptions(const char * arg, int num_options, cups_option_t
** options);
```

Description

Parse options from a command-line argument.

This function converts space-delimited name/value pairs according to the PAPI text option ABNF specification. Collection values ("name={a=... b=... c=...}") are stored with the curley brackets intact - use cupsParseOptions() on the value to extract the collection attributes.

Return Value

Number of options found

cupsPrintFile

Name

`cupsPrintFile` — Print a file to a printer or class on the default server.

Synopsis

```
#include <cups/cups.h>
int cupsPrintFile(const char * name, const char * filename, const
char * title, int num_options, cups_option_t * options);
```

Description

Print a file to a printer or class on the default server.

Return Value

Job ID

cupsPrintFiles

Name

`cupsPrintFiles` — Print one or more files to a printer or class on the

Synopsis

```
#include <cups/cups.h>
int cupsPrintFiles(const char * name, int num_files, const char **
files, const char * title, int num_options, cups_option_t *
options);
```

Description

Print one or more files to a printer or class on the default server.

Return Value

Job ID

cupsServer

Name

`cupsServer` — Return the hostname/address of the default server.

Synopsis

```
#include <cups/cups.h>
const char * cupsServer(void);
```

Description

Return the hostname/address of the default server.

The returned value can be a fully-qualified hostname, a numeric IPv4 or IPv6 address, or a domain socket pathname.

Return Value

Server name

cupsSetDests

Name

`cupsSetDests` — Save the list of destinations for the default server.

Synopsis

```
#include <cups/cups.h>
void cupsSetDests(int num_dests, cups_dest_t * dests);
```

Description

Save the list of destinations for the default server.

This function saves the destinations to /etc/cups/lpoptions when run as root and ~/.cups/lpoptions when run as a normal user.

Return Value

This function does not return a value.

cupsSetEncryption

Name

`cupsSetEncryption` — Set the encryption preference.

Synopsis

```
#include <cups/cups.h>
void cupsSetEncryption(http_encryption_t e);
```

Description

Set the encryption preference.

Return Value

This function does not return a value.

cupsSetPasswordCB

Name

`cupsSetPasswordCB` — Set the password callback for CUPS.

Synopsis

```
#include <cups/cups.h>
void cupsSetPasswordCB(cups_password_cb_t cb);
```

Description

Set the password callback for CUPS.

Pass NULL to restore the default (console) password callback.

Return Value

This function does not return a value.

cupsSetServer

Name

`cupsSetServer` — Set the default server name.

Synopsis

```
#include <cups/cups.h>
void cupsSetServer(const char * server);
```

Description

Set the default server name.

The "server" string can be a fully-qualified hostname, a numeric IPv4 or IPv6 address, or a domain socket pathname. Pass NULL to restore the default server name.

Return Value

This function does not return a value.

cupsSetUser

Name

`cupsSetUser` — Set the default user name.

Synopsis

```
#include <cups/cups.h>
void cupsSetUser(const char * user);
```

Description

Set the default user name.

Pass NULL to restore the default user name.

Return Value

This function does not return a value.

cupsTempFd

Name

`cupsTempFd` — Creates a temporary file.

Synopsis

```
#include <cups/cups.h>
int cupsTempFd(char * filename, int len);
```

Description

Creates a temporary file.

The temporary filename is returned in the filename buffer. The temporary file is opened for reading and writing.

Return Value

New file descriptor or -1 on error

cupsUser

Name

`cupsUser` — Return the current user's name.

Synopsis

```
#include <cups/cups.h>
const char * cupsUser(void);
```

Description

Return the current user's name.

Return Value

User name

ppdClose

Name

`ppdClose` — Free all memory used by the PPD file.

Synopsis

```
#include <cups/cups.h>
void ppdClose(ppd_file_t * ppd);
```

Description

Free all memory used by the PPD file.

Return Value

This function does not return a value.

ppdCollect

Name

`ppdCollect` — Collect all marked options that reside in the specified

Synopsis

```
#include <cups/cups.h>
int    ppdCollect(ppd_file_t    *    ppd,    ppd_section_t    section,
ppd_choice_t *** choices);
```

Description

Collect all marked options that reside in the specified section.

Return Value

Number of options marked

ppdConflicts

Name

`ppdConflicts` — Check to see if there are any conflicts.

Synopsis

```
#include <cups/cups.h>
int ppdConflicts(ppd_file_t * ppd);
```

Description

Check to see if there are any conflicts.

Return Value

Number of conflicts found

ppdEmit

Name

`ppdEmit` — Emit code for marked options to a file.

Synopsis

```
#include <cups/cups.h>
int ppdEmit(ppd_file_t * ppd, FILE * fp, ppd_section_t section);
```

Description

Emit code for marked options to a file.

Return Value

0 on success, -1 on failure

ppdEmitFd

Name

`ppdEmitFd` — Emit code for marked options to a file.

Synopsis

```
#include <cups/cups.h>
int ppdEmitFd(ppd_file_t * ppd, int fd, ppd_section_t section);
```

Description

Emit code for marked options to a file.

Return Value

0 on success, -1 on failure

ppdEmitJCL

Name

`ppdEmitJCL` — Emit code for JCL options to a file.

Synopsis

```
#include <cups/cups.h>
int ppdEmitJCL(ppd_file_t * ppd, FILE * fp, int job_id, const char *
user, const char * title);
```

Description

Emit code for JCL options to a file.

Return Value

0 on success, -1 on failure

ppdErrorString

Name

`ppdErrorString` — Returns the text assocated with a status.

Synopsis

```
#include <cups/cups.h>
const char * ppdErrorString(ppd_status_t status);
```

Description

Returns the text assocated with a status.

Return Value

Status string

ppdFindAttr

Name

`ppdFindAttr` — Find the first matching attribute...

Synopsis

```
#include <cups/cups.h>
ppd_attr_t * ppdFindAttr(ppd_file_t * ppd, const char * name, const
char * spec);
```

Description

Find the first matching attribute...

Return Value

Attribute or NULL if not found

ppdFindChoice

Name

`ppdFindChoice` — Return a pointer to an option choice.

Synopsis

```
#include <cups/cups.h>
ppd_choice_t * ppdFindChoice(ppd_option_t * o, const char *
choice);
```

Description

Return a pointer to an option choice.

Return Value

Choice pointer or NULL

ppdFindMarkedChoice

Name

`ppdFindMarkedChoice` — Return the marked choice for the specified option.

Synopsis

```
#include <cups/cups.h>
ppd_choice_t * ppdFindMarkedChoice(ppd_file_t * ppd, const char *
option);
```

Description

Return the marked choice for the specified option.

Return Value

Pointer to choice or NULL

ppdFindNextAttr

Name

`ppdFindNextAttr` — Find the next matching attribute...

Synopsis

```
#include <cups/cups.h>
ppd_attr_t * ppdFindNextAttr(ppd_file_t * ppd, const char * name,
const char * spec);
```

Description

Find the next matching attribute...

Return Value

Attribute or NULL if not found

ppdFindOption

Name

`ppdFindOption` — Return a pointer to the specified option.

Synopsis

```
#include <cups/cups.h>
ppd_option_t * ppdFindOption(ppd_file_t * ppd, const char * option);
```

Description

Return a pointer to the specified option.

Return Value

Pointer to option or NULL

ppdIsMarked

Name

`ppdIsMarked` — Check to see if an option is marked...

Synopsis

```
#include <cups/cups.h>
int ppdIsMarked(ppd_file_t * ppd, const char * option, const char *
choice);
```

Description

Check to see if an option is marked...

Return Value

Non-zero if option is marked

ppdLastError

Name

`ppdLastError` — Return the status from the last ppdOpen*().

Synopsis

```
#include <cups/cups.h>
ppd_status_t ppdLastError(int * line);
```

Description

Return the status from the last ppdOpen*().

Return Value

Status code

ppdMarkDefaults

Name

`ppdMarkDefaults` — Mark all default options in the PPD file.

Synopsis

```
#include <cups/cups.h>
void ppdMarkDefaults(ppd_file_t * ppd);
```

Description

Mark all default options in the PPD file.

Return Value

This function does not return a value.

ppdMarkOption

Name

`ppdMarkOption` — Mark an option in a PPD file.

Synopsis

```
#include <cups/cups.h>
int ppdMarkOption(ppd_file_t * ppd, const char * option, const char
* choice);
```

Description

Mark an option in a PPD file.

Notes:

-1 is returned if the given option would conflict with any currently selected option.

Return Value

Number of conflicts

ppdOpen

Name

`ppdOpen` — Read a PPD file into memory.

Synopsis

```
#include <cups/cups.h>
ppd_file_t * ppdOpen(FILE * fp);
```

Description

Read a PPD file into memory.

Return Value

PPD file record

ppdOpenFd

Name

`ppdOpenFd` — Read a PPD file into memory.

Synopsis

```
#include <cups/cups.h>
ppd_file_t * ppdOpenFd(int fd);
```

Description

Read a PPD file into memory.

Return Value

PPD file record

ppdOpenFile

Name

`ppdOpenFile` — Read a PPD file into memory.

Synopsis

```
#include <cups/cups.h>
ppd_file_t * ppdOpenFile(const char * filename);
```

Description

Read a PPD file into memory.

Return Value

PPD file record

ppdPageLength

Name

`ppdPageLength` — Get the page length for the given size.

Synopsis

```
#include <cups/cups.h>
float ppdPageLength(ppd_file_t * ppd, const char * name);
```

Description

Get the page length for the given size.

Return Value

Length of page in points or 0.0

ppdPageSize

Name

`ppdPageSize` — Get the page size record for the given size.

Synopsis

```
#include <cups/cups.h>
ppd_size_t * ppdPageSize(ppd_file_t * ppd, const char * name);
```

Description

Get the page size record for the given size.

Return Value

Size record for page or NULL

ppdPageWidth

Name

`ppdPageWidth` — Get the page width for the given size.

Synopsis

```
#include <cups/cups.h>
float ppdPageWidth(ppd_file_t * ppd, const char * name);
```

Description

Get the page width for the given size.

Return Value

Width of page in points or 0.0

ppdSetConformance

Name

`ppdSetConformance` — Set the conformance level for PPD files.

Synopsis

```
#include <cups/cups.h>
void ppdSetConformance(ppd_conform_t c);
```

Description

Set the conformance level for PPD files.

Return Value

This function does not return a value.

7.4 Interfaces for libcupsimage

Table 7-3 defines the library name and shared object name for the libcupsimage library

Table 7-3 libcupsimage Definition

Library:	libcupsimage
SONAME:	libcupsimage.so.2

The behavior of the interfaces in this library is specified by the following specifications:

[LSB] This Specification

7.4.1 CUPS Raster ABI

7.4.1.1 Interfaces for CUPS Raster ABI

An LSB conforming implementation shall provide the generic functions for CUPS Raster ABI specified in Table 7-4, with the full mandatory functionality as described in the referenced underlying specification.

Table 7-4 libcupsimage - CUPS Raster ABI Function Interfaces

cupsRasterClose [LSB]	cupsRasterOpen [LSB]	cupsRasterReadHeader [LSB]	cupsRasterReadPixels [LSB]
cupsRasterWriteHeader [LSB]	cupsRasterWritePixels [LSB]		

7.5 Data Definitions for libcupsimage

This section defines global identifiers and their values that are associated with interfaces contained in libcupsimage. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the ISO C (1999) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

7.5.1 cups/raster.h

```
#define _CUPS_RASTER_H_
#define CUPS_RASTER_SYNC        0x52615374
#define CUPS_RASTER_REVSYNC     0x74536152
#define CUPS_RASTER_HAVE_COLORIMETRIC   1

typedef enum {
    CUPS_RASTER_READ = 0,
    CUPS_RASTER_WRITE = 1
} cups_mode_t;
typedef struct {
    unsigned int sync;
    int fd;
    cups_mode_t mode;
} cups_raster_t;
typedef enum {
    CUPS_ADVANCE_NONE = 0,
    CUPS_ADVANCE_FILE = 1,
    CUPS_ADVANCE_JOB = 2,
    CUPS_ADVANCE_SET = 3,
    CUPS_ADVANCE_PAGE = 4
} cups_adv_t;
typedef enum {
    CUPS_FALSE = 0,
    CUPS_TRUE = 1
} cups_bool_t;
typedef enum {
    CUPS_CUT_NONE = 0,
    CUPS_CUT_FILE = 1,
    CUPS_CUT_JOB = 2,
    CUPS_CUT_SET = 3,
    CUPS_CUT_PAGE = 4
} cups_cut_t;
typedef enum {
    CUPS_JOG_NONE = 0,
    CUPS_JOG_FILE = 1,
    CUPS_JOG_JOB = 2,
    CUPS_JOG_SET = 3
} cups_jog_t;
typedef enum {
    CUPS_EDGE_TOP = 0,
    CUPS_EDGE_RIGHT = 1,
    CUPS_EDGE_BOTTOM = 2,
    CUPS_EDGE_LEFT = 3
} cups_edge_t;
typedef enum {
    CUPS_ORIENT_0 = 0,
    CUPS_ORIENT_90 = 1,
    CUPS_ORIENT_180 = 2,
    CUPS_ORIENT_270 = 3
} cups_orient_t;
```

```
typedef enum {
    CUPS_ORDER_CHUNKED = 0,
    CUPS_ORDER_BANDED = 1,
    CUPS_ORDER_PLANAR = 2
} cups_order_t;
typedef enum {
    CUPS_CSPACE_W = 0,
    CUPS_CSPACE_RGB = 1,
    CUPS_CSPACE_RGBA = 2,
    CUPS_CSPACE_K = 3,
    CUPS_CSPACE_CMY = 4,
    CUPS_CSPACE_YMC = 5,
    CUPS_CSPACE_CMYK = 6,
    CUPS_CSPACE_YMCK = 7,
    CUPS_CSPACE_KCMY = 8,
    CUPS_CSPACE_KCMYcm = 9,
    CUPS_CSPACE_GMCK = 10,
    CUPS_CSPACE_GMCS = 11,
    CUPS_CSPACE_WHITE = 12,
    CUPS_CSPACE_GOLD = 13,
    CUPS_CSPACE_SILVER = 14,
    CUPS_CSPACE_CIEXYZ = 15,
    CUPS_CSPACE_CIELab = 16,
    CUPS_CSPACE_ICC1 = 32,
    CUPS_CSPACE_ICC2 = 33,
    CUPS_CSPACE_ICC3 = 34,
    CUPS_CSPACE_ICC4 = 35,
    CUPS_CSPACE_ICC5 = 36,
    CUPS_CSPACE_ICC6 = 37,
    CUPS_CSPACE_ICC7 = 38,
    CUPS_CSPACE_ICC8 = 39,
    CUPS_CSPACE_ICC9 = 40,
    CUPS_CSPACE_ICCA = 41,
    CUPS_CSPACE_ICCB = 42,
    CUPS_CSPACE_ICCC = 43,
    CUPS_CSPACE_ICCD = 44,
    CUPS_CSPACE_ICCE = 45,
    CUPS_CSPACE_ICCF = 46
} cups_cspace_t;
typedef struct {
    char MediaClass[64];
    char MediaColor[64];
    char MediaType[64];
    char OutputType[64];
    unsigned int AdvanceDistance;
    cups_adv_t AdvanceMedia;
    cups_bool_t Collate;
    cups_cut_t CutMedia;
    cups_bool_t Duplex;
    unsigned int HWResolution[2];
    unsigned int ImagingBoundingBox[4];
    cups_bool_t InsertSheet;
    cups_jog_t Jog;
    cups_edge_t LeadingEdge;
    unsigned int Margins[2];
    cups_bool_t ManualFeed;
    unsigned int MediaPosition;
    unsigned int MediaWeight;
    cups_bool_t MirrorPrint;
    cups_bool_t NegativePrint;
    unsigned int NumCopies;
    cups_orient_t Orientation;
    cups_bool_t OutputFaceUp;
    unsigned int PageSize[2];
    cups_bool_t Separations;
    cups_bool_t TraySwitch;
    cups_bool_t Tumble;
    unsigned int cupsWidth;
    unsigned int cupsHeight;
    unsigned int cupsMediaType;
    unsigned int cupsBitsPerColor;
    unsigned int cupsBitsPerPixel;
    unsigned int cupsBytesPerLine;
    cups_order_t cupsColorOrder;
    cups_cspace_t cupsColorSpace;
    unsigned int cupsCompression;
    unsigned int cupsRowCount;
    unsigned int cupsRowFeed;
    unsigned int cupsRowStep;
} cups_page_header_t;
extern void cupsRasterClose(cups_raster_t *);
extern   unsigned   int   cupsRasterWritePixels(cups_raster_t   *,
unsigned char *,
                                          unsigned int);
extern unsigned int cupsRasterReadHeader(cups_raster_t *,
                                         cups_page_header_t *);
extern   unsigned   int   cupsRasterReadPixels(cups_raster_t   *,
unsigned char *,
                                         unsigned int);
extern cups_raster_t *cupsRasterOpen(int, cups_mode_t);
extern unsigned int cupsRasterWriteHeader(cups_raster_t *,
                                          cups_page_header_t *);
```

7.6 Interface Definitions for libcupsimage

The interfaces defined on the following pages are included in libcupsimage and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed in Section 7.4 shall behave as described in the referenced base document.

cupsRasterClose

Name

`cupsRasterClose` — Close a raster stream.

Synopsis

```
#include <cups/raster.h>
void cupsRasterClose(cups_raster_t * r);
```

Description

Close a raster stream.

Return Value

This function does not return a value.

cupsRasterOpen

Name

`cupsRasterOpen` — Open a raster stream.

Synopsis

```
#include <cups/raster.h>
cups_raster_t * cupsRasterOpen(int fd, cups_mode_t mode);
```

Description

Open a raster stream.

Return Value

New stream

cupsRasterReadHeader

Name

`cupsRasterReadHeader` — Read a raster page header and store it in a

Synopsis

```
#include <cups/raster.h>
unsigned cupsRasterReadHeader(cups_raster_t * r, cups_page_header_t
* h);
```

Description

Read a raster page header and store it in a V1 page header structure.

Return Value

1 on success, 0 on fail

cupsRasterReadPixels

Name

`cupsRasterReadPixels` — Read raster pixels.

Synopsis

```
#include <cups/raster.h>
unsigned cupsRasterReadPixels(cups_raster_t * r, unsigned char * p,
unsigned len);
```

Description

Read raster pixels.

Return Value

Number of bytes read

cupsRasterWriteHeader

Name

`cupsRasterWriteHeader` — Write a raster page header from a V1 page

Synopsis

```
#include <cups/raster.h>
unsigned cupsRasterWriteHeader(cups_raster_t * r, cups_page_header_t
* h);
```

Description

Write a raster page header from a V1 page header structure.

Return Value

1 on success, 0 on failure

cupsRasterWritePixels

Name

`cupsRasterWritePixels` — Write raster pixels.

Synopsis

```
#include <cups/raster.h>
unsigned cupsRasterWritePixels(cups_raster_t * r, unsigned char * p,
unsigned len);
```

Description

Write raster pixels.

Return Value

Number of bytes written

III Printing Commands

8 Printing Commands

8.1 Printing Commands

An LSB conforming implementation shall provide the commands and utilities as described in Table 8-1, with at least the behavior described as mandatory in the referenced underlying specification, with the following exceptions:

1. If any operand (except one which follows `--`) starts with a hyphen, the behavior is unspecified.

Rationale (Informative): Applications should place options before operands, or use `--`, as needed. This text is needed because, by default, GNU option parsing differs from POSIX, unless the environment variable POSIXLY_CORRECT is set. For example, **ls . -a** in GNU **ls** means to list the current directory, showing all files (that is, `"."` is an operand and `-a` is an option). In POSIX, `"."` and `-a` are both operands, and the command means to list the current directory, and also the file named `-a`. Suggesting that applications rely on the setting of the POSIXLY_CORRECT environment variable, or try to set it, seems worse than just asking the applications to invoke commands in ways which work with either the POSIX or GNU behaviors.

Table 8-1 Commands And Utilities

foomatic-rip [1]	gs [1]			

Referenced Specification(s)

[1]. This Specification

8.2 Command Behavior

This section contains descriptions for commands and utilities whose specified behavior in the LSB contradicts or extends the standards referenced. It also contains commands and utilities only required by the LSB and not specified by other standards.

FOOMATIC-RIP

2002-11-26

Name

`foomatic-rip` — Universal print filter/RIP wrapper

SYNOPSIS

Standalone Mode

foomatic-rip [-v] [-q] [-d] [--ppd ppdfile] [-J jobtitle] [-o option=value [...]] [files]

CUPS Mode

foomatic-rip jobid user jobtitle numcopies options [file]

DESCRIPTION

foomatic-rip is a universal print filter which works with every known free software printer spooler.

This page describes the facilities of foomatic-rip when used as a CUPS filter and when used outside of a print system. While implementations of foomatic-rip may support other print systems, such use is not documented here.

When run as a CUPS filter, foomatic-rip reads the job from the specified file, or from standard input if no file is specified. It renders the file into a printer-specific format, and writes the result to standard output.

When run standalone, foomatic-rip will read the job from the specified files, or from standard input if no files are given. The files are rendered into a printer-specific format, which is then output according to the PPD option "FoomaticRIPPostPipe", documented in the LSB.

Printer capabilities are described to foomatic-rip via PPD files, as described (with extensions used by foomatic-rip) in the LSB. The method foomatic-rip uses to determine the proper PPD file for the printer in question is defined by the implementation of both the spooler and foomatic-rip.

CUPS OPTIONS

Unless otherwise noted, all parameters are required when running foomatic-rip as a CUPS filter.

`jobid`

The internal Job ID from CUPS.

`username`

The username of the user who submitted the job.

`jobtitle`

The job's title, as submitted by the user.

`numcopies`

The number of copies of the job requested.

`options`

A series of printer options, separated by spaces, each of which take the form `name` or `name=value`. The specific list of options supported is dependent on the printer and spooler, and is usually documented in the PPD file for the printer.

An option may be preceded by a page specification, describing the pages to which the option should apply. A page specification consists of one or more items, separated by commas, and separated from the option name by a colon. Valid items include the words "even" and "odd", a single page number, and a page range. Page ranges are described with a starting page, a dash ("-"), and an ending page. If omitted, the starting and ending pages are the first and last page, respectively, but only one of the ends of the range may be omitted.

`file`

The full path to the file containing the job. This parameter is optional; if it is not supplied, the job is read from standard input.

SPOOLER-LESS OPTIONS

`-v`

Verbose mode. Intended for debugging and testing purposes only.

`-q`

Quiet mode - minimal information output.

`-d`

Identical to the 'opts' option, but option information is left in text format. The PPD file will need to be specified using the --ppd option.

`--ppd` *`ppdfile`*

The PPD file `ppdfile` should be applied for processing this job.

`-J` *`jobtitle`*

Print the given job title in the header of every page of a plain text job.

`-o` *`option=value`*

Set an option setting for this job.

EXIT STATUS

foomatic-rip returns 0 unless something unexpected happens.

AUTHOR

Till Kamppeter <*till.kamppeter@gmail.com*> with parts of Manfred Wassmanns's <*manolo@NCC-1701.B.Shuttle.de*> man pages for the Foomatic 2.0.x filters.

Jeff Licquia <*licquia@linux-foundation.org*> adapted the original man page for the LSB.

GS

2007-11-29

Name

`gs` — GhostScript (PostScript and PDF language interpreter)

SYNOPSIS

gs -h | --help

gs [options] ps-file [[options] ps-file2] ...

DESCRIPTION

The gs command invokes Ghostscript, an interpreter of Adobe Systems' PostScript(tm) and Portable Document Format (PDF) languages. gs reads the files named by ps-file in sequence and executes them as Ghostscript programs. After doing this, it reads further input from the standard input stream (normally the keyboard), interpreting each line separately. The interpreter exits gracefully when it encounters the "quit" command (either in a file or from the keyboard), at end-of-file, or at an interrupt signal (such as Control-C at the keyboard).

Some of GhostScript's options are set via command-line options; others are set as processing parameters, each of which consists of a name and a value.

OPTIONS

`-h --help`

Show GhostScript's help, as well as lists of the supported input formats, supported devices, and the search path for gs components.

`-q`

Suppress normal startup messages, and also set the processing parameter QUIET.

`-c`

Begin interpreting arguments as PostScript code. All following arguments are sent to the interpreter up to the next argument beginning with "-" followed by a non-digit, or with "@". This code is interpolated with the file list, so files specified before *`-c`* are interpreted beforehand, and files after *`-c`* are interpreted afterwards.

`-f`

Specifies a PostScript file to run as its argument. This is equivalent to the ps-file arguments, but is useful for terminating PostScript code as passed via *`-c`*.

`-d -D`

Set a processing parameter. The "name=value" pair follows immediately after the option, as in "-Dfoo=bar". The values here must be integers or the values "true" or "false". The equals sign and value may be omitted; this is assumed to set the name to "true".

`-s -S`

Set a processing parameter to a string value. The "name=value" pair follows immediately after the option, as in "-Sfoo=bar".

-u

Unset a processing parameter. The name to be unset follows immediately after the option, as in "-ufoo".

-o

Write rendered output to the named file, and also inhibit pauses and the interactive shell. This is equivalent to setting the processing parameters BATCH and NOPAUSE to true, and OutputFile to the parameter after -o.

-r

Set the device resolution. The resolution is specified as two numbers separated with an "x", as in "300x150", corresponding to the X and Y axis resolutions, respectively. If a single number is given without an "x", it is treated as the value for both resolutions.

This is equivalent to setting DEVICEXRESOLUTION and DEVICEYRESOLUTION in systemdict.

-g

Set the device size, in pixels. The size is specified as two numbers separated with an "x", as in "640x480", corresponding to the width and height, respectively.

This is equivalent to setting DEVICEWIDTH and DEVICEHEIGHT in systemdict.

RECOGNIZED PROCESSING PARAMETERS

Processing parameters may have arbitrary names; no limits are placed on the settings that may be made. However, certain settings have meaning to the gs interpreter, and drivers may use other settings. Below is a list of recognized settings that the gs interpreter must respect.

BATCH

If set to true, do not enter an interactive shell after processing all command-line files.

DEVICE

Contains the name of the device used to render the page, as a string.

The list of available devices can be discovered with the -h parameter, as described above. At least the following devices must be present: cups (CUPS Raster), ijs, pxlmono, pxlcolor, and opvp (OpenPrinting Vector).

DEVICEHEIGHT

Contains the height, in pixels, of the output device.

The effect of this setting when the current driver is a vector driver is undefined.

DEVICEHEIGHTPOINTS

Sets the initial page height, in units of 1/72 of an inch.

DEVICEWIDTH

Contains the width, in pixels, of the output device.

The effect of this setting when the current driver is a vector driver is undefined.

DEVICEWIDTHPOINTS

Sets the initial page width, in units of 1/72 of an inch.

DEVICEXRESOLUTION

Contains the resolution, in pixels per inch, of the X dimension (horizontal) of the output device.

DEVICEYRESOLUTION

Contains the resolution, in pixels per inch, of the Y dimension (vertical) of the output device.

NOPAUSE

If set to true, disable the prompt and pause normally displayed after rendering a page.

OutputFile

Contains the path to the file to which gs should write its output, as a string. This parameter may be set to '-', in which case gs's output is sent to standard output.

PAPERSIZE

Contains the string representation of the paper size. The ISO paper sizes a0-a10 (plus a4small), isob0-isob6, and c0-c6 are recognized, as are jisb0-jisb6 (JIS standard sizes) and the US paper sizes 11x17, ledger, legal, letter, lettersmall, and archA-archE.

QUIET

If set to true, suppress routine information comments on standard output.

SAFER

If set to true, disable several unsafe PostScript features: the deletefile and renamefile operators, piped commands, reading or writing to general files, and changing of certain system settings.

STRICT

If set to true, disable as many extensions to the Adobe PostScript specification as possible.

EXIT STATUS

gs returns 0 on successful execution. Any other return value indicates an error.

AUTHOR

Jeff Licquia (licquia@linux-foundation.org) wrote this man page for the LSB specification.

Portions of this page were taken from the GhostScript documentation. The maintainer and rights holder for GhostScript is Artifex Software, Inc.

Annex A GNU Free Documentation License (Informative)

This specification is published under the terms of the GNU Free Documentation License, Version 1.1, March 2000

A.1 PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

A.2 APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A.3 VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

A.4 COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

A.5 MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

A.6 COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

A.7 COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

A.8 AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

A.9 TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

A.10 TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

A.11 FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

A.12 How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.