X Print Service

Protocol Version 1.0

X Consortium Standard
X Version 11, Release 6.4

A. Deininger
T. Gilg
J. Miller
H. Phinney
C. Prince

Hewlett-Packard Co.

K. Samborn
R. Swick

X Consortium, Inc.
Copyright (c) 1996 Hewlett-Packard Company
Copyright (c) 1996 International Business Machines, Inc.
Copyright (c) 1996 Sun Microsystems, Inc.
Copyright (c) 1996 Novell, Inc.
Copyright (c) 1996 Digital Equipment Corp.
Copyright (c) 1996 Fujitsu Limited
Copyright (c) 1996 Hitachi, Ltd.
Copyright (c) 1996 X Consortium, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE X CONSORTIUM BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of the X Consortium shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from the X Consortium.

*X Window System* is a trademark of X Consortium, Inc.
# Table of Contents

1 Overview ................................................................................................................................................... 1

2 Protocols ................................................................................................................................................... 2  
   2.1 Formats, Syntactic Conventions, and Common Types ................................................................. 2  
   2.2 Errors ........................................................................................................................................... 2  
   2.3 X Print Service Requests ............................................................................................................. 2  
   2.4 Events ......................................................................................................................................... 11

3 X Print Attributes .................................................................................................................................... 13  
   3.1 Attribute Value Defaults And Validation .................................................................................... 13  
   3.2 Server Attributes ....................................................................................................................... 14  
   3.3 Printer Attributes ....................................................................................................................... 14  
   3.4 Job Attributes ............................................................................................................................ 17  
   3.5 Document Attributes .................................................................................................................. 17  
   3.6 Page Attributes .......................................................................................................................... 18

4 Communication with the Print Dialog Manager ..................................................................................... 20

5 Protocol Encoding ................................................................................................................................... 22  
   5.1 Request Protocol Encoding ......................................................................................................... 22  
   5.2 Event Protocol Encoding ............................................................................................................. 31  
   5.3 Error Protocol Encoding ............................................................................................................. 32
Overview

1 Overview

X Print Service is an X extension that allows X imaging to non-display devices such as printers and fax machines. The core of the X Print Service is the X Print Server.

Applications that require printing operations can make a connection to X Print Server and list the available printers using the `PrintGetPrinterList` request. After selecting a printer, an application must create and set a print context using the `PrintCreateContext` and `PrintSetContext` requests.

The “print context” is a fundamental X Print Service concept. A print context:

- Contains a printer’s default capabilities
- Contains a printer’s range of capabilities
- Maintains the state of the settings on a printer
- Maintains the state of rendering against a printer
- Maintains rendered output

A print context also affects how the DDX driver generates its page description language (PDL), and how the PDL is submitted to a spooler. It may affect fonts and other elements in the DDX layer of the X Print Server.

Printer capabilities are defined by attribute pools within the print context. They contain information related to a context’s server, printer, job, document, and page options. `PrintGetAttributes` and `PrintSetAttributes` are used to access and modify attribute pools.

`PrintStartJob` and `PrintEndJob` are used to delineate print jobs. A “job” is a collection of documents delineated by `PrintStartDoc` and `PrintEndDoc`. Each document is, in turn, a collection of “pages”. Upon completion the server sends any resulting PDL to a print spooler, or makes it available for retrieval by an application.
2 Protocols

2.1 Formats, Syntactic Conventions, and Common Types

The type PCONTEXT is a 32-bit value. Its top three bits are guaranteed to be zero.

Refer to the X Window System Protocol specification for a description of other formats, syntactic conventions and common types established in that publication and used in the current document as well.

2.2 Errors

X Print Service can return the following messages, in addition to X core request errors.

XPBadContext  An incorrect print context ID was specified.

XPBadSequence  Requests were not specified in the proper order with respect to other requests. For example, a request was specified before a PrintSetContext request.

Other errors that are context specific for a particular request are documented in the description of the request itself. If the above errors have a specific meaning for a particular request, they are documented in the request itself as well.

2.3 X Print Service Requests

PrintCreateContext

context-id: PCONTEXT
printer-name: STRING8
locale: STRING8

Errors: Match, IDChoice

This request creates a new print context and assigns context-id to it. The attributes associated with the new context are those determined by the printer-name. Printer-name is encoded in COMPOUND_TEXT.

The client must select the context-id by ORing some combination of bits in the connection resource-id-mask with the resource-id-base.

The locale argument is used as a “hint” to the print server, and is used to initialize attribute pools with any localized attribute values.

A Match error is generated when the printer-name does not exist.
PrintSetContext

context: PCONTEXT or None

Errors: XPBadContext

This request associates the context specified with all subsequent print operations for this client. If context is None, the print context previously associated with this client is unset. If no print context was previously set, then no action is taken when None is specified.

The execution of the PrintSetContext request may affect the interpretation of the font path. The font path contains font path elements for all printers associated with a print server. Only those associated with the current print context are returned and used for print rendering.

PrintGetContext

→

context: PCONTEXT or None

This request returns the current print context for the connection.

PrintDestroyContext

context: PCONTEXT

Errors: XPBadContext

This request unsets and destroys a print context. If a print context is destroyed before print operations associated with it have been completed, the print server cancels all those operations as if a CancelJob request had been issued.

PrintGetPrinterList

printer-name: STRING8
locale: STRING8

→

printers: LISTofPRINTER
where:
    PRINTER: name: STRING8
description: STRING8

This request retrieves a list of all printers supported on a print server.
If printer-name is an empty string, then a list of all printers is returned. Otherwise the print record that matches the printer-name specified is returned. If no records match printer-name, then an empty list is returned.

printer-name is a COMPOUND_TEXT string. The name and description fields returned are COMPOUND_TEXT. If printer-name is provided in a code-set that the print server cannot convert, then it may not be possible to locate the requested printer.

The locale argument is used as a “hint” to locate a localized description for each printer in the list. If the print server cannot interpret the hint, then it describes the printers in the server’s current locale.

---

**PrintGetScreenOfContext**

→

root: WINDOW

Errors: **XPBadContext**

This request returns the root window associated with the current print context.

Each printer supported by a print server is associated with exactly one of the screens returned in the connection setup reply.

---

**PrintStartJob**

output-mode: \{XPSpool, XPGetData\}

Errors: **XPBadSequence, Value**

This request signals the beginning of a new print job. It results in the generation of an **XPPrintNotify** event, with the detail field set to **XPStartJobNotify**.

If output-mode is set to **XPSpool**, then the document data is typically sent to a spooler.

If output-mode is set to **XPGetData**, then the document data is made available to **PrintGetDocumentData** and the resulting job is not spooled. In this case, the print server suspends processing further requests on this print context until some other client sends **PrintGetDocumentData**. Subsequent operations that use the print context may be suspended at any time pending the processing of **PrintGetDocumentData** replies to read any buffered output.

Any changes to the **XPJobAttr** pool must be made before **PrintStartJob**. Further modifications can only be made to the attribute pool after a **PrintEndJob** request is executed.

---

**PrintEndJob**

cancel: BOOL
Errors: XPBadContext, XPBadSequence

This request causes the print job associated with the current print context to end. If cancel is FALSE, any accumulated print data that remains is either sent to the printer or made available to PrintGetDocumentData.

The request generates an XPPrintNotify event with its detail field set to XPEndJobNotify.

When cancel is TRUE, the job currently being processed is canceled. The server may discard any pending output or may produce partial output. If the job was started in XPEndJobData mode, then the entire data output stream is implementation-defined.

If PrintEndJob is called immediately after PrintEndPage, then a synthetic PrintEndDoc is generated by print server before PrintEndJob. The pool of XPJobAttr attributes that was frozen when the PrintStartJob request was executed is released when PrintEndJob is called.

PrintGetDocumentData

context: PCONTEXT
max-bytes: CARD32

status-code: {XPGetDocFinished, XPGetDocSecondConsumer, XPGetDocError}
finished-flag: CARD32
data: LISTofBYTE

Errors: XPBadContext, XPBadSequence, Value

This request returns data generated on a context by other clients.

PrintGetDocumentData should be sent only after a PrintStartJob request with save_data set to XPGData has been executed.

PrintGetDocumentData generates multiple replies. Each reply is no larger than the value specified in max-bytes. The final reply is generated by PrintEndJob and has finished-flag set to TRUE.

If the value for max-bytes is zero, a Value error is generated.

An XPBadSequence error is generated if PrintGetDocumentData is executed before PrintStartJob or if PrintGetDocumentData is executed after PrintStartJob with save_data set to XPSpool.

PrintPutDocumentData

drawable: DRAWABLE
data: LISTofBYTE
doc-format, options: STRING8

Errors: XPBadContext, XPBadSequence, Match, Value, Drawable
This request allows an application to send and incorporate data into the print output. It functions in two modes, depending on whether the PrintStartDoc driver-mode is set to XPDocNormal or XPDocRaw:

**XPDocNormal**

PrintPutDocumentData sends data to the print server and integrates data into the output. The root of the drawable must be the root of the current print context. The doc-format and options parameters describe the format of data, which in turn guides the way the server interprets it. The **xp-embedded-formats-supported** attribute in the XPPrinterAttr pool defines valid values for doc-format in this mode, else a Match error is issued.

**XPDocRaw**

PrintPutDocumentData sends data directly to the print server output. The print server does not emit document or page control codes into the output, and data is passed through unmodified. Drawable must be None, else a Drawable error is issued. The **xp-raw-formats-supported** attribute in the XPPrinterAttr pool defines valid values for doc-format in this mode, else a Match error is issued.

If doc-format is not in **xp-embedded-formats-supported** or **xp-raw-formats-supported** a Value error is issued. The options field is implementation-dependent and the permitted values may depend on the current settings of other attributes and the value of doc-format. If an unknown options value is specified a Value error is issued, else if options is not valid in the current state a Match error is issued.

---

**PrintStartDoc**

driver-mode: \{XPDocNormal, XPDocRaw\}

Errors: Value, XPBadSequence

This request indicates the beginning of an individual document within a print job. The server performs the actions necessary to define a new document, and generates an XPPrintNotify event with its detail field set to XPStartDocNotify.

The value of driver-mode can be:

**XPDocNormal**

Print server generates document data. Depending on the DDX driver, it can incorporate data from PrintPutDocumentData into the document.

**XPDocRaw**

The client provides all data for the document using PrintPutDocumentData. The print server does not generate any data of its own into the document.

If PrintStartPage is sent immediately after PrintStartJob, then a synthetic PrintStartDoc with driver-mode XPDocNormal will be generated internally by print server before PrintStartPage.

Any changes to the XPDocAttr attribute pool must be made before PrintStartDoc is executed. Further modifications can only be made to the attribute pool after a PrintEndDoc request is executed.

---

**PrintEndDoc**
cancel: BOOL

Errors: XPBadSequence

This request signals the end of a print document. The resulting document data is assembled and combined with data that was sent by \texttt{PrintPutDocumentData}.

When cancel is \texttt{TRUE}, the document currently being processed is canceled. The server may discard any pending output or may produce partial output. If the job was started with \texttt{XPGetData} mode, then the entire data output stream is implementation-defined for this document.

The \texttt{XpDocAttr} pool that was frozen when the \texttt{PrintStartDoc} request was executed is released when \texttt{PrintEndDoc} is called.

\textbf{PrintStartPage}

\texttt{window: WINDOW}

Errors: XPBadSequence, Window

This request indicates the beginning of a single print page within a document. Window is the drawable that represents the page.

\texttt{PrintStartPage} causes window to be mapped. Within a \texttt{PrintStartPage/PrintEndPage} sequence, any attempts to resize, move, or unmap window will be ignored. To resize or move inferiors of window, the standard semantics used for \texttt{ConfigureWindow} apply, except that the contents of the configured window may be lost. If the contents of a window are lost, an \texttt{Expose} event is generated.

A \texttt{Window} error is issued if window is not a descendent of the root window of the current print context. An \texttt{XPBadSequence} error is issued if \texttt{PrintStartPage} is called in an \texttt{XPDocRaw} document.

Any changes to the \texttt{XPPageAttr} attribute pool must be made before \texttt{PrintStartPage} is executed. Further modifications can only be made to the attribute pool after a \texttt{PrintEndPage} request is executed.

\textbf{PrintEndPage}

cancel: BOOL

Errors: XPBadContext, XPBadSequence

This request indicates the end of a print page, and causes window to be unmapped. If cancel is \texttt{TRUE}, the current print page is canceled.

When cancel is \texttt{TRUE}, the job currently being processed is canceled. The server may discard any pending output or may produce partial output. If the job was started with \texttt{XPGetData} mode, then the entire data output stream is implementation-defined for this page.

The pool of job attributes that was frozen when the \texttt{PrintStartPage} request was executed is freed when \texttt{PrintEndPage} is called.
**PrintGetPageDimensions**

color: PCONTEXT

text: width: CARD16
text: height: CARD16
text: offset-x: CARD16
text: offset-y: CARD16
text: reproducible-width: CARD16
text: reproducible-height: CARD16

Errors: XPBadContext

This request returns the total width and height of a page in pixels, together with the net reproducible area within the page. The net reproducible area is the portion of the page on which the printer is physically capable of placing ink.

**PrintSelectInput**

color: PCONTEXT

text: event-mask: BITMASK

Errors: XPBadContext, Value

This request specifies the print events, from those in the specified print context, the client is interested in. Possible values for the event-mask BITMASK are:

- XPNoEventMask
- XPPrintMask
- XPAtributeMask

**PrintInputSelected**

color: PCONTEXT

Errors: XPBadContext
This request queries which X Print Server events the client has selected to receive from the specified print context. all-events-mask returns the set of all events selected by all clients.

PrintGetAttributes
context: PCONTEXT
pool: {XPJobAttr, XPDocAttr, XPageAttr, XPrinterAttr, XServerAttr}

→ attributes: STRING8

Errors: XBadContext, Value

This request returns an attribute pool from the specified print context. attributes is the attribute pool specified by pool, and is encoded in COMPOUND_TEXT.

The format used for attributes is the same as the format used for an X resource file. For a description see Section 15.1, “Resource File Syntax”, in the Xlib specification.

See section 3 for a detailed description of attributes.

PrintGetOneAttribute
context: PCONTEXT
pool: {XPJobAttr, XPDocAttr, XPageAttr, XPrinterAttr, XServerAttr}
name: STRING8

→ value: STRING8

Errors: XBadContext, Value

This request retrieves a single attribute from the specified print context. It is similar to PrintGetAttributes, but returns only one attribute value instead of an entire pool of attributes. The specific attribute is specified by name. value is encoded in COMPOUND_TEXT.

PrintSetAttributes
context: PCONTEXT
pool: {XPJobAttr, XPDocAttr, XPageAttr, XPrinterAttr, XServerAttr}
rule: {XAttrMerge, XAttrReplace}
attributes: STRING8

Errors: XBadContext, XBadSequence, Value, Match
This request sets the names and values for one or more attributes within the specified attribute pool. attributes is encoded in COMPOUND_TEXT that represents new name/value pairs according to the value specified in rule. For **XPAttrReplace**, the existing attribute pool is discarded and replaced with attributes. For **XPAttrMerge**, attributes is merged into the existing attribute pool; existing name/value pairs are replaced and new ones are added.

The format used for attributes is the same as the format used for an X resource file. For a description see Section 15.1, “Resource File Syntax”, in the Xlib specification.

See section 3 for a detailed description of attributes.

A **Match** message is returned if read-only attribute pools attempt to use **PrintSetAttributes**. An **XPBadSequence** message is issued when a request is sent to an attribute pool at a time when the attribute pool cannot be modified.

### PrintRehashPrinterList

This request causes the print server to update its list of available printers together with their attributes. The print facilities underlying X Server may provide it with the ability to detect changes in printer topology and dynamically update the list to reflect the changes. If X Server does not have this capability, the **PrintRehashPrinterList** must be used to notify it of changes.

### PrintQueryVersion

→

major-version, minor-version: CARD16

This request returns the major and minor version numbers of the X Print Service.

### PrintQueryScreens

→

roots: LISTofWINDOW

This request returns a list of the X Server screens that support the X Print Service Extension.

### PrintSetImageResolution

context: PCONTEXT
image-resolution: CARD16
→
status: BOOL
previous-resolution: CARD16

Errors: XPBadContext

This request sets the resolution for subsequent PutImage requests on the screen of context, in pixels per inch.

When status is TRUE, then the contents of any subsequent PutImage request to a Pixmap or to a Window on the screen of the specified print context will automatically be scaled as part of the PutImage request. The scale factor is:

\[
\text{default-printer-resolution} / \text{image-resolution}
\]

where default-printer-resolution is the current value of that page attribute. Only the image itself is scaled (meaning the effective width and height of the image change), the dst-x and dst-y parameters to PutImage are not altered.

As a special case, a value of zero for image_res resets the resolution to automatically track the printer resolution. In this case (which is also the default setting for a newly created print context), subsequent images will not be scaled.

previous-resolution is the previous image resolution that was set for context in pixels per inch.

If status is FALSE, then the print server does not support image scaling for the particular resolution given the current configuration of the printer, and the application is responsible for any desired scaling.

PrintGetImageResolution

context: PCONTEXT
→
image-resolution: CARD16

Errors: XPBadContext

This request returns the current image-resolution for context in pixels per inch. A value of zero means the resolution automatically tracks the printer resolution. If the request fails in some way, a negative value is returned.

2.4 Events

XPPrintNotify

detail: [XPStartJobNotify, XPEndJobNotify, XPStartDocNotify, XPEndDocNotify, XPStartPageNotify, XPEndPageNotify]

cancel: BOOL

context: PCONTEXT
This event is generated when requests to `PrintStartDoc`, `PrintStartJob`, `PrintStartPage`, `PrintEndDoc`, `PrintEndJob`, and `PrintEndPage` have been processed and completed. It is reported to clients selecting `XPPrintMask`.

```
XPAttributeNotify
detail: {XPJobAttr, XPDocAttr, XPageAttr, XPPrinterAttr, XPServerAttr, XPMediumAttr, XPSpoolerAttr}
context: PCONTEXT
```

This event is generated when any of the print attribute pools maintained by the print server have been modified. The modifications may have been initiated by the print server itself or by a `PrintSetAttributes` request. It is reported to clients selecting `XPAttributeMask`.
3 X Print Attributes

Printing-specific attributes play a key role in the X Print Service. They provide a general-purpose mecha-
nism for storing information associated with printing. This information includes user print setup options,
printer capabilities, and spooler subsystem options.

The X Print Service selects attributes in a way that is consistent with the X Windows System, ISO/IEC
10175 (ISO DPA), and POSIX 1387.4 print standards. The ISO DPA defines a number of abstract objects
that are managed and manipulated during the printing process. These are known as DPA-Objects. Each
DPA-Object is represented by a set of attributes which characterize that object. Each attribute in turn is com-
posed of an attribute-type (attribute name) and zero or more attribute-values.

The X Print Service utilizes selected DPA-Objects, and for each of these, a subset of the associated
attributes. The DPA-Objects used are:

Server Object Specifies attributes defined for the X print server.

Job Object Specifies attributes for a single print request as sent to the spooler.

Document Object Specifies attributes used to define a single document within a job. If supported by
the implementation, multiple documents may be submitted within a given job.

Printer Object Specifies attributes that identify printer capabilities.

The X Print Service also provides for changing certain attributes on a page-by-page basis. This is a
capability for which the ISO DPA does not define a separate DPA-Object. This set of attributes is known
within the X Print Service as Page Attributes.

The X Print Service requires some additional attributes that are not defined by the ISO DPA. The attribute
names for these attributes are prefixed with "xp-".

A server implementation can define additional attributes.

This section defines the following sets of attributes for the X Print Service:

• Server Attributes
• Printer Attributes
• Job Attributes
• Document Attributes
• Page Attributes

3.1 Attribute Value Defaults And Validation

This section provides an overview of the handling of default attribute values and the procedure for the vali-
dation of attribute values within the X Print Service. Details for individual attributes can be found in the rest
of this chapter.

3.1.1 Assigning Attribute Value Defaults

An attribute specification with an empty value indicates that the attribute has no value. Within X Print Ser-
vice configuration files and attribute pools, an attribute specification that omits the value is effectively
treated as if there were no attribute specification. An empty valued attribute specification that has precedence
over a non-empty attribute specification (for instance, an empty printer qualified attribute over a non-empty
model qualified attribute) will effectively “unset” the lower precedence attribute specification. When a print job commences, the X Print Service may infer a default value for an attribute that has no value. In some cases the X Print Service may explicitly assign a default value to an attribute before presenting it in an attribute pool.

### 3.1.2 Validating Attribute Values

The X Print Server ensures that attribute pools presented to the client are always comprised of valid attribute specifications for attributes defined by the X Print Service. Validation is first performed when a print context is created. Validation is also performed whenever a client requests an update to an attribute pool.

Validation involves checking the attribute value against its set of valid values. The process may also take into account the current values of other attributes and the capabilities of the DDX driver.

Attributes may be single-valued or multi-valued.

When a print context is created, if the server determines that an attribute value is invalid, the server will ignore the invalid attribute specification and may set an explicit default for the attribute in the pool. For multi-valued attributes, the server will ignore each value component that is invalid. If all of the specified components are invalid the server will reject the attribute specification, and for certain attributes will set an explicit default for the attribute in the pool.

When the client requests an update to an attribute pool (e.g. when issuing `PrintSetAttributes`), if the server determines that a single-valued attribute is invalid, that attribute will not be updated. If all components of a multi-valued attribute are invalid the attribute will not be updated, otherwise any invalid components are ignored. Unrecognized attributes will be stored in the corresponding attribute pool and returned in `PrintGetAttribute`, but are otherwise ignored.

As part of the validation for a given attribute, the print server may alter other attributes in response to the change. For example, changing the value of the `document-format` attribute might cause the value of the `xp-embedded-formats-supported` attribute to change as.

### 3.1.3 Structured Values

### 3.2 Server Attributes

The server attribute pool is identified by `XPServerAttr` and describes the capabilities of the X Print Server.

- **locale**
  The value of this attribute is the locale in which the X Print Server is running.

- **multiple-documents-supported**
  This attribute indicates whether the server supports jobs containing multiple documents.

### 3.3 Printer Attributes

The printer attribute pool is identified by `XPPrinterAttr` and describes printer capabilities.
content-orientations-supported
A list of orientations supported in the print context. The list is a group of strings separated by white space. Valid values are portrait, landscape, reverse-portrait, and reverse-landscape.

The default value is determined by the DDX, and is explicitly set in the printer pool. Validation for this attribute is as described for multi-valued attributes in “Validating Attribute Values”.

The initial value of the content-orientations-supported attribute is typically set by the printer vendor in the model-config file.

descriptor
The descriptor is a human readable description of the printer encoded as COMPOUND_TEXT. This description may contain more than one line.

document-attributes-supported
A list of document attributes supported in the print context. This list is returned as a set of whitespace-delimited attribute names.

document-formats-supported
A list of document formats, including format variants and format versions that are supported in the print context. Each entry in the list is a structure comprised of the document-format, document-format-variant, and a document-format-version. Variant and version may be omitted in some cases. The triple value is enclosed by curly braces “{}” and delimited by whitespace.

input-trays-medium
This attribute identifies what medium is loaded in each printer tray. The value is specified as a list of structures, each of which contains a tray identifier and a medium identifier. Valid tray identifiers are top, middle, bottom, envelope, manual, large-capacity, main, and side. The X Print Service defines valid medium identifiers to be the standard values of the medium-size attribute as specified in ISO/IEC 10175-1.

For each tray / medium (size) combination, the tray must be present in the value of the medium-source-sizes-supported attribute, and the medium size must be listed for that tray.

job-attributes-supported
A list of the job attributes supported for the printer. This list is returned as a set of whitespace-delimited attribute names.

medium-source-sizes-supported
This attribute identifies or specifies the sizes of media that are supported by the printer. For each input tray a set of supported media sizes is indicated. For each medium, the page size, an indicator as to the medium feed direction, and the assured reproduction area the printer supports are specified.

Valid input tray values are top, middle, bottom, envelope, manual, large-capacity, main, and side. If the printer has only one input tray, specification of this value is optional.

The page size is a descriptive-name indicating the size of the page. Examples are iso-a4, na-letter, and na-legal. The complete list of valid values is the set of descriptive-names defined for the standard values of the medium-size attribute as specified in ISO/IEC 10175-1.
The medium feed direction is represented as a boolean value indicating whether the long edge (TRUE) or the short edge (FALSE) feeds into the printer so that orientation is specified.

The assured reproduction area is the area within the current medium to which the printer can render. This area is specified in millimeters according to the RCS coordinate system defined by the ISO DPA. The area value is defined by a structure containing the minimum-x, maximum-x, minimum-y, and maximum-y.

The value for a medium size is specified in a structure comprised of the page size, the feed direction indicator, and the assured reproduction area.

The value of the medium-source-sizes-supported attribute is a list of structures, each comprised of the input tray value and a set of medium size values.

plexes-supported A list of plex options that the printer supports. The list is a group of strings separated by white space. Valid values are simplex, duplex, and tumble.

printer-model Human-readable text that identifies the make and model of the printer. This value is encoded as COMPOUND_TEXT.

printer-name This attribute uniquely identifies a printer on a given X Print Server.

printer-resolutions-supported A list of the resolutions in dots per inch that the printer supports.

xp-embedded-formats-supported This attribute identifies the set of data formats recognized as valid values for the doc_fmt parameter of PrintPutDocumentData, when this request is issued within a print document of type XPDocNormal.

The value is a list of data formats. Each entry in the list is a structure comprised of the data format, a format variant, and a format version. The variant and the version may be omitted in some cases. Structure values are enclosed by curly braces “{}” and delimited by whitespace. Valid values are defined by the printer DDX driver.

xp-listfonts-modes-supported Defines the set of values that may be used to comprise the value of the xp-listfonts-modes document / page attribute. The value is a whitespace delimited list of listfonts mode values, which are defined below.

xp-page-attributes-supported A list of page attributes supported for the printer. This list is comprised of a set of whitespace-delimited attribute names.

xp-raw-formats-supported This attribute identifies the set of data formats recognized as valid values for the doc_fmt parameter of PrintPutDocumentData, when this function is called within a print document of type XPDocRaw.

The value is a list of data formats. Each entry in the list is a structure comprised of the data format, an optional format variant, and an optional format version. Structure values are enclosed by curly braces “{}” and delimited by whitespace. Valid values are defined based on the physical printer’s capabilities.
xp-setup-proviso

This attribute indicates whether or not a required attribute or set of attributes must be set prior to commencing the print job.

Valid values for this attribute are xp-setup-mandatory and xp-setup-optional. If this attribute is not specified, xp-setup-optional is assumed.

The initial value of the xp-setup-proviso attribute is typically set by the printer vendor in the model-config file.

### 3.4 Job Attributes

The job attribute pool is identified by XPJobAttr and provides information on how to process a print job. Typically, job attributes are set by the Print Dialog Manager based on user input from the setup dialog.

**job-name**

This is the name of the job to be used in subsequent processing and in printing banner pages. The value is free form text.

**job-owner**

This attribute identifies the human owner of the print job.

**notification-profile**

This attribute is a specification of events about which the user is to be notified. The X Print service uses this attribute to determine whether or not to notify the user of print job completion via electronic mail, or in ISO DPA parlance, the X Print Service recognizes the event-report-job-completed event with a delivery-method of electronic-mail.

The values may be {{event-report-job-completed} electronic-mail} to send an email message, and {} if no message is to be sent. Servers may implement additional values.

**xp-setup-state**

If the value of the xp-setup-proviso printer attribute is xp-setup-mandatory, then xp-setup-state is used to indicate the current setup state as determined by X Print Server. If the value of xp-setup-proviso is xp-setup-optional, the value of xp-setup-state is ignored.

Valid values for xp-setup-state are xp-setup-ok and xp-setup-incomplete. xp-setup-ok indicates that all attributes the print server requires the user to set are valid, indicating a client may commence printing if desired. xp-setup-incomplete indicates that one or more attributes the driver requires are unspecified or invalid; printing should not be attempted.

**xp-spooler-command-options**

A free form text string that will be included verbatim on the command line used to invoke the spooler. Valid values are spooler-dependent.

**xp-spooler-command-results**

A free form text string that will contain the spooler command output that would otherwise appear on a terminal (e.g. stderr and stdout). This text may be useful to present to the user to allow tracking of the resulting spooler job. Applications should retrieve this value following receipt of the XPEndJobNotify event.

### 3.5 Document Attributes

The document attribute pool is identified by XPDocAttr and indicates how to process the current document.
content-orientation  Specifies the orientation to be used for this document. Valid values are: portrait, landscape, reverse-portrait, and reverse-landscape.

copy-count  Specifies the number of copies of this document to print.

The default value is implicitly taken to be 1 by the X Print Server.

default-printer-resolution  Specifies the resolution in dots per inch to be used for this document.

default-input-tray  The name of the input tray from which media will be drawn for printing the document. Valid values are: top, middle, bottom, envelope, manual, large-capacity, main, and side. If the default-medium attribute is specified, it will take precedence over default-input-tray.

default-medium  Specifies the medium on which the document is to be printed. The X Print Service defines valid default-medium values to be the standard values of the medium-size attribute as specified in ISO/IEC 10175-1.

document-format  Specifies the format of the document. The value is a structure comprised of the document-format, an optional document-format-variant, and an optional document-format-version. Specific printer DDX drivers may require specification of the optional values. The structure values are enclosed by curly braces “{}” and delimited by whitespace.

plex  Specifies the plex to be used for this document. Valid values are simplex, duplex, and tumble.

xp-listfonts-modes  The value of this attribute controls the behavior of ListFonts and ListFontsWithInfo when a print context has been set. The value is a whitespace delimited list of one or more listfonts mode values. Valid listfonts mode values include xp-list-internal-printer-fonts and xp-list-glyph-fonts.

When a print context is set on a display connection, the default behavior of ListFonts and ListFontsWithInfo is to list all of the fonts normally associated with the X print server (i.e. fonts containing glyphs) as well as any internal printer fonts defined for the printer. The xp-listfonts-modes attribute is provided so that applications can control the behavior of ListFonts and ListFontsWithInfo and is typically to show just internal printer fonts. Using only internal printer fonts is useful for performance reasons; the glyphs associated with the font are contained within the printer and do not have to be downloaded to it.

If the value of xp-listfonts-modes includes xp-list-glyph-fonts, ListFonts and ListFontsWithInfo will include all of the fonts available to the server which have glyphs associated with them. If the value of xp-listfonts-modes includes xp-list-internal-printer-fonts, then ListFonts and ListFontsWithInfo will include all of the fonts defined as internal printer fonts.

3.6  Page Attributes

The page attribute pool is identified by XPPageAttr. These are document attributes that can be overridden on a page by page basis within the X Print Service.

The default for each page attribute is the current value of the corresponding document attribute.
**content-orientation** Specifies the orientation to be used for this page. Valid values are: portrait, landscape, reverse-portrait, and reverse-landscape.

**default-printer-resolution**
Specifies the resolution in dots per inch to be used for this page.

**default-input-tray**
The name of the input tray from which media will be drawn for printing the document. Valid values are: top, middle, bottom, envelope, manual, large-capacity, main, and side. If the default-medium attribute is specified, it will take precedence over default-input-tray.

**default-medium**
Specifies the medium on which the document is to be printed. The X Print Service defines valid default-medium values to be the standard values of the medium-size attribute as specified in ISO/IEC 10175-1.

**plex**
Specifies the plex to be used for this document. Valid values are simplex, duplex, and tumble.

**xp-listfonts-modes**
The value of this attribute controls the behavior of ListFonts and ListFontsWithInfo when a print context has been set. The value is a whitespace delimited list of one or more listfonts mode values. Valid listfonts mode values include xp-list-internal-printer-fonts and xp-list-glyph-fonts.

When a print context is set on a display connection, the default behavior of ListFonts and ListFontsWithInfo is to list all of the fonts normally associated with the X print server (i.e. fonts containing glyphs) as well as any internal printer fonts defined for the printer. The xp-listfonts-modes attribute is provided so that applications can control the behavior of ListFonts and ListFontsWithInfo and is typically to show just internal printer fonts. Using only internal printer fonts is useful for performance reasons; the glyphs associated with the font are contained within the printer and do not have to be downloaded to it.

If the value of xp-listfonts-modes includes xp-list-glyph-fonts, ListFonts and ListFontsWithInfo will include all of the fonts available to the server which have glyphs associated with them. If the value of xp-listfonts-modes includes xp-list-internal-printer-fonts, then ListFonts and ListFontsWithInfo will include all of the fonts defined as internal printer fonts.
4 Communication with the Print Dialog Manager

Print Dialog Managers (PDMs) provide users with a graphical interface to specify printer-specific and spooler-specific information. This section describes the interaction between X Print Service and PDMs.

For each server wanting to use print dialog services, a Print Dialog Manager acquires ownership of a selection named PDM_MANAGER on the default root window (a different name can be used, as long as it is known to both the client and the PDM). Print Dialog Managers should comply with the conventions for “Manager Selections” described in section 2.8 of the Inter-Client Communication Conventions Manual (ICCCM). A printing client establishes a print context, and then requests services of the Print Dialog Manager by issuing conversion requests on this selection.

Print Dialog Managers should support conversion of the following targets on their manager selection:

<table>
<thead>
<tr>
<th>ATOM</th>
<th>DATA RECEIVED</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDM_START</td>
<td>Request that a dialog be managed for a particular print context</td>
</tr>
</tbody>
</table>

The PDM_START Selection Target

The PDM_START target is parametrized (ICCCM section 2.2), and the property named in the ConvertSelection request contains the following list of information:

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>FORMAT</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>video-display</td>
<td>“host:port[.screen]”</td>
<td>X display of video server</td>
</tr>
<tr>
<td>video-window</td>
<td>“0x12345678”</td>
<td>Window to act as parent of PDM dialog</td>
</tr>
<tr>
<td>print-display</td>
<td>“host:port[.screen]”</td>
<td>X display of print server</td>
</tr>
<tr>
<td>print-window</td>
<td>“0x12345678”</td>
<td>Window on print server for subsequent communication</td>
</tr>
<tr>
<td>print-context</td>
<td>“0x12345678”</td>
<td>Context of print job</td>
</tr>
<tr>
<td>locale</td>
<td>“C”</td>
<td>Hint to PDM regarding the locale</td>
</tr>
</tbody>
</table>

The PDM_START target has a side effect (ICCCM, section 2.6.3). The PDM interprets the parameters listed above and provides a user interface dialog on behalf of the client in which the user can modify attributes on the print-context on the print-display provided.

The PDM will use the video-display, video-window, and locale parameters to configure and manage its user interface.

The selection reply is placed in the property provided, where type is ATOM, format is 32, and the data consists of a single ATOM element:

<table>
<thead>
<tr>
<th>ATOM</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDM_START_OK</td>
<td>The PDM was started successfully</td>
</tr>
<tr>
<td>PDM_START_VXAUTH</td>
<td>The PDM was not authorized to connect to video-display</td>
</tr>
</tbody>
</table>
If the PDM starts successfully, once the user completes the PDM dialog, the PDM finishes communication with the client by sending a ClientMessage to print-window on the print-display.

The type of this ClientMessage is “PDM_REPLY”, its format is 32, and the data consists of a single ATOM element:

<table>
<thead>
<tr>
<th>ATOM</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDM_EXIT_OK</td>
<td>The user selected “OK”. The PDM may or may not have changed any attributes.</td>
</tr>
<tr>
<td>PDM_EXIT_CANCEL</td>
<td>The user selected “Cancel”. Attributes have been left in the state they were in before communication began.</td>
</tr>
<tr>
<td>PDM_EXIT_VXAUTH</td>
<td>The PDM was not authorized to connect to video-display</td>
</tr>
<tr>
<td>PDM_EXIT_PXAUTH</td>
<td>The PDM was not authorized to connect to print-display</td>
</tr>
<tr>
<td>PDM_EXIT_ERROR</td>
<td>The PDM encountered an error</td>
</tr>
</tbody>
</table>
5 Protocol Encoding

The following sections describe protocol encoding for X Print Extension Protocol requests, events, and errors.

5.1 Request Protocol Encoding

PrintQueryVersion

1 base major im
1 0 minor opcode
2 1 request length

→
1 1 Reply
1 unused
2 CARD16 sequence number
4 0 reply length
2 CARD16 major-version
2 CARD16 minor-version
20 unused

PrintGetPrinterList

1 base major opcode
1 1 minor opcode
2 3+(nl+np + ll+lp)/4 request length
4 CARD32 printerNameLen
4 CARD32 localeLen
nl STRING8 printer-name
np BYTE p=pad(nl)
ll STRING8 locale
lp BYTE lp=pad(ll)

→
1 1 Reply
1 unused
2 CARD16 sequenceNumber
Protocol Encoding

4 (8 + nl+nlp + dl+dlp)/4 computed length
listCount times
4 CARD32 listCount
20 unused
(8 + nl+nlp + dl+dlp) computed listCount times
LISTofPRINTER printers

PRINTERS
4 CARD32 nameLen
nl STRING8 name
nlp BYTE nlp=pad(nl)
4 CARD32 descLen
dl STRING8 description
dl p BYTE dlp=pad(dl)

PrintRehashPrinterList

1 base major opcode
1 20 minor opcode
2 1 request length

PrintCreateContext

1 base major opcode
1 2 minor opcode
2 4 + (nl+np + ll+lp)/4 request length
4 CARD32 context-id
4 CARD32 printerNameLen
4 CARD32 localeLen
nl STRING8 printer-name
np STRING8 np=pad(nl)
ll STRING8 locale
lp BYTE lp=pad(ll)
Protocol Encoding

**PrintSetContext**

1  base  major opcode
1  3    minor opcode
2  2    request length
4  CARD32 context

**PrintGetContext**

1  base  major opcode
1  4    minor opcode
2  1    request length

→

1  1    Reply
1  unused
2  CARD16 sequence number
4  0    reply length
4  CARD32 context
16 unused

**PrintDestroyContext**

1  base  major opcode
1  5    minor opcode
2  2    request length
4  CARD32 context

**PrintGetScreenofContext**

1  base  major opcode
1  6    minor opcode
2  1    request length

→
### Protocol Encoding

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Reply</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>unused</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CARD16</td>
<td>sequence number</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>reply length</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>WINDOW</td>
<td>root</td>
<td>unused</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### PrintStartJob

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>base</td>
<td>major opcode</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>minor opcode</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>request length</td>
</tr>
<tr>
<td>1</td>
<td>CARD8</td>
<td>output-mode</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>unused</td>
</tr>
</tbody>
</table>

#### PrintEndJob

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>base</td>
<td>major opcode</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>minor opcode</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>request length</td>
</tr>
<tr>
<td>1</td>
<td>BOOL</td>
<td>cancel</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>unused</td>
</tr>
</tbody>
</table>

#### PrintStartDoc

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>base</td>
<td>major opcode</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>minor opcode</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>request length</td>
</tr>
<tr>
<td>1</td>
<td>CARD8</td>
<td>driver-mode</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>unused</td>
</tr>
</tbody>
</table>
Protocol Encoding

PrintEndDoc

1 base major opcode
1 10 minor opcode
2 2 request length
1 BOOL cancel
3 unused

PrintPutDocumentData

1 base major opcode
1 11 minor opcode
2 4 + (d+dp + f+fp + o+op)/4 request length
4 DRAWABLE drawable
4 CARD32 len_data
2 CARD16 len_fmt
2 CARD16 len_options
4 LISTofBYTE data
4 STRING8 doc-format
4 STRING8 options
4 STRING8 options
4 STRING8 options

PrintGetDocumentData

1 base major opcode
1 12 minor opcode
2 3 request length
4 PCONTEXT context
4 CARD32 max-bytes

1 Reply
1 unused
2 CARD16 sequence number
4 (n + p)/4 reply length
4 0 XPGetDocFinished status-code
1 XPGetDocSecondConsumer
4 CARD32 finished-flag
4 CARD32 dataLen
Protocol Encoding

PrintStartPage

12 \text{ ListOfBYTE} \quad \text{unused}
\n1 \text{ BYTE} \quad \text{data}
\np = \text{pad}(n)

PrintStartPage

1 \text{ base} \quad \text{major opcode}
\n1 13 \quad \text{minor opcode}
\n2 2 \quad \text{request length}
\n4 \text{ WINDOW} \quad \text{window}

PrintEndPage

1 \text{ base} \quad \text{major opcode}
\n1 14 \quad \text{minor opcode}
\n2 2 \quad \text{request length}
\n1 \text{ BOOL} \quad \text{cancel}
\n3 \text{ unused}

PrintSelectInput

1 \text{ base} \quad \text{major opcode}
\n1 15 \quad \text{minor opcode}
\n2 3 \quad \text{request length}
\n4 \text{ PCONTEXT} \quad \text{context}
\n4 \text{ BITMASK} \quad \text{event-mask}
\#x00000000 \quad \text{XPNoEventMask}
\#x00000001 \quad \text{XPPrintMask}
\#x00000002 \quad \text{XPAtributeMask}

PrintInputSelected

1 \text{ base} \quad \text{major opcode}
\n1 16 \quad \text{minor opcode}
28 Protocol Encoding

2 2 request length
4 PCONTEXT context

→

1 1 Reply
1 unused
2 CARD16 sequence number
4 0 reply length
4 BITMASK event-mask
4 BITMASK all-events-mask
16 unused

PrintGetAttributes

1 base major opcode
1 17 minor opcode
2 3 request length
4 PCONTEXT context
1 CARD8 pool
3 unused

→

1 1 Reply
1 unused
2 CARD16 sequence number
4 (n+p)/4 reply length
4 CARD32 stringLen
20 unused
n STRING8 attributes
p p=pad(n)

PrintGetOneAttribute

1 base major opcode
1 19 minor opcode
2 4 + (n+p)/4 request length
4 PCONTEXT context
4 CARD32 nameLen
1 CARD8 pool
Protocol Encoding

3
n STRING8
p unused

→

1 1 Reply
1 unused
2 CARD16 sequence number
4 (n+p)/4 reply length
4 CARD32 valueLen
20 unused
n STRING8 value
p unused

PrintSetAttributes

1 base major opcode
1 18 minor opcode
2 4 + (n+p)/4 request length
4 PCONTEXT context
4 CARD32 stringLen
1 CARD8 pool
1 CARD8 rule
2 unused
n STRING8 attributes
p unused

PrintGetPageDimensions

1 base major opcode
1 21 minor opcode
2 2 request length
4 PCONTEXT context

→

1 1 Reply
1 unused
2 CARD16 sequence number
4 0 reply length
Protocol Encoding

PrintQueryScreens

1 base major opcode
1 22 minor opcode
2 2 request length

→

1 1 Reply
1 unused
2 CARD16 sequence number
4 listCount reply length
4 CARD32 listCount
20 unused
4 * listCount LISTofWINDOW roots

ROOT-WINDOW
4 WINDOW rootWindow

PrintSetImageResolution

1 base major opcode
1 23 minor opcode
2 3 request length
4 PCONTEXT context
4 CARD16 image-resolution
2 unused

→

1 1 Reply
1 BOOL status
Protocol Encoding

PrintGetImageResolution

1  base  major opcode
1  24  minor opcode
2  2  request length
4  PCONTEXT  context

→

1  1  Reply
1  unused
2  CARD16  sequence number
4  0  reply length
2  CARD16  image-resolution
22  unused

5.2 Event Protocol Encoding

PrintNotify

1  0 + base  code
1  0 XPStartJobNotify  detail
1 XPEndJobNotify
2 XPStartDocNotify
3 XPEndDocNotify
4 XPStartPageNotify
5 XPEndPageNotify
2  CARD16  sequence number
4  PCONTEXT  context
1  BOOL  cancel
23  unused
AttributeNotify

1  1 + base code
1  1 XPJobAttr detail
2  XPDocAttr
3  XPPageAttr
4  XPPrinterAttr
5  XPServerAttr
6  XPMediumAttr (future use)
7  XPSpoolerAttr (future use)

2  CARD16 sequence number
4  PCONTEXT context
24 unused

5.3 Error Protocol Encoding

BadContext

1  0 Error
1  0 + base code
2  CARD16 sequence number

BadSequence

1  0 Error
1  1 + base code
2  CARD16 sequence number
A
AttributeNotify, encoding 32
attributes 13–19
defaults 13
document 17
job 17
page 18
printer 14
server 14
validating 14

B
BadContext, encoding 32
BadSequence, encoding 32

C
content-orientation 18, 19
content-orientations-supported 15
copy-count 18

D
default-input-tray 18, 19
default-medium 18, 19
default-printer-resolution 18, 19
defaults, attributes 13
descriptor 15
document attributes 17
document-attributes-supported 15
document-format 18
document-formats-supported 15

E
errors 2

I
input-trays-medium 15

J
job attributes 17
job-attributes-supported 15
job-name 17
job-owner 17

L
locale, attribute 14

M
medium-source-sizes-supported 15
multiple-documents-supported 14

N
notification-profile 17

P
page attributes 18
PCONTEXT, type 2
plex 18, 19
plexes-supported 16
Print Dialog Manager, communicating with 20
PrintCreateContext
encoding 23
request 2
PrintDestroyContext
encoding 24
request 3
PrintEndDoc
encoding 25
request 4
PrintEndJob
encoding 26
request 6
PrintEndPage
encoding 27
request 7
printer attributes 14
printer-model 16
printer-name 16
printer-resolutions-supported 16
PrintGetAttributes
encoding 28
request 9
PrintGetContext
encoding 24
request 3
PrintGetDocumentData
encoding 26
request 5
PrintGetImageResolution
encoding 31
request 11
PrintGetOneAttribute
encoding 28
request 9
PrintGetPageDimensions
encoding 29
request 8
PrintGetPrinterList
Index

encoding 22
request 3
PrintGetScreenOfContext
encoding 24
request 4
PrintInputSelected
encoding 27
request 8
PrintNotify, encoding 31
PrintPutDocumentData
encoding 26
request 5
PrintQueryScreens
encoding 30
request 10
PrintQueryVersion
encoding 22
request 10
PrintRehashPrinterList
encoding 23
request 10
PrintSelectInput
encoding 27
request 8
PrintSetAttributes
encoding 29
request 9, 10
PrintSetContext
encoding 24
request 3
PrintSetImageResolution
encoding 30
request 10
PrintStartDoc
encoding 25
request 6
PrintStartJob
encoding 25
request 4
PrintStartPage
encoding 27
request 7

S
server attributes 14

V
validating attributes 14

X
XPAttributeNotify

event 12
XPBadContext, error description 2
XPBadSequence, error description 2
xp-embedded-formats-supported 16
xp-listfonts-modes 18, 19
xp-listfonts-modes-supported 16
xp-page-attributes-supported 16
XPPrintNotify
event 11
xp-raw-formats-supported 16
xp-setup-proviso 17
xp-setup-state 17
xp-spooler-command-options 17
xp-spooler-command-results 17