EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

WORKSHOP AGREEMENT

CWA 14050-17

November 2000

ICS 35.200; 35.240.40

Extensions for Financial Services (XFS) interface specification - Release 3.0 - Part 17: Printer Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Ref. No CWA 14050-17:2000 E

Rue de Stassart, 36 • B-1050 Bruxelles
Tel : +32 2 550 08 11 • Fax : +32 2 550 08 19

Table of Contents

Foreword

This CWA is revision 3.0 of the XFS interface specification.

The move from an XFS 2.0 specification (CWA 13449) to a 3.0 specification has been prompted by a series of factors.

Initially, there has been a technical imperative to extend the scope of the existing specification of the XFS Manager to include new devices, such as the Card Embossing Unit.

Similarly, there has also been pressure, through implementation experience and the advance of the Microsoft technology, to extend the functionality and capabilities of the existing devices covered by the specification.

Finally, it is also clear that our customers and the market are asking for an update to a specification, which is now over 2 years old. Increasing market acceptance and the need to meet this demand is driving the Workshop towards this release.

The clear direction of the CEN/ISSS XFS Workshop, therefore, is the delivery of a new Release 3.0 specification based on a C API. It will be delivered with the promise of the protection of technical investment for existing applications and the design to safeguard future developments.

The CEN/ISSS XFS Workshop gathers suppliers as well as banks and other financial service companies. A list of companies participating in this Workshop and in support of this CWA is available from the CEN/ISSS Secretariat.

This CWA was formally approved by the XFS Workshop meeting on 2000-10-18. The specification is continuously reviewed and commented in the CEN/ISSS Workshop on XFS. It is therefore expected that an update of the specification will be published in due time as a CWA, superseding this revision 3.0.

The CWA is published as a multi-part document, consisting of:

Part 1: Application Programming Interface (API) - Service Provider Interface (SPI); Programmer's Reference

Part 2: Service Classes Definition; Programmer's Reference

Part 3: Printer Device Class Interface - Programmer's Reference

Part 4: Identification Card Device Class Interface - Programmer's Reference

Part 5: Cash Dispenser Device Class Interface - Programmer's Reference

Part 6: PIN Keypad Device Class Interface - Programmer's Reference

Part 7: Check Reader/Scanner Device Class Interface - Programmer's Reference

Part 8: Depository Device Class Interface - Programmer's Reference

Part 9: Text Terminal Unit Device Class Interface - Programmer's Reference

Part 10: Sensors and Indicators Unit Device Class Interface - Programmer's Reference

Part 11: Vendor Dependent Mode Device Class Interface - Programmer's Reference

Part 12: Camera Device Class Interface - Programmer's Reference

Part 13: Alarm Device Class Interface - Programmer's Reference

Part 14: Card Embossing Unit Class Interface - Programmer's Reference

Part 15: Cash In Module Device Class Interface- Programmer's Reference

Part 16: Application Programming Interface (API) - Service Provider Interface (SPI) - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 17: Printer Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 18: Identification Card Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 19: Cash Dispenser Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 20: PIN Keypad Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 21: Depository Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 22: Text Terminal Unit Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 23: Sensors and Indicators Unit Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 24: Camera Device Class Interface - Migration from Version 2.0 (see CWA 13449) to Version 3.0 (this CWA) - Programmer's Reference

Part 25: Identification Card Device Class Interface - PC/SC Integration Guidelines

In addition to these Programmer's Reference specifications, the reader of this CWA is also referred to a complementary document, called Release Notes. The Release Notes contain clarifications and explanations on the CWA specifications, which are not requiring functional changes. The current version of the Release Notes is available online from http://www.cenorm.be/isss/Workshop/XFS.

1. General

A new type scanner, multiple supplies, multiple retract bins and UNICODE support for FieldValues have been added. In addition to that the command for scanning images has also been improved.

2. New Chapters

2.1. References

1. XFS Application Programming Interface (API)/Service Provider Interface (SPI), Programmer's Reference Revision 3.00, October 18, 2000

2.2. XFS form/media definition files in multi-vendor environments

Although for most Service Providers directory location and extension of XFS form/media definition files are configurable through the registry, the capabilities of Service Providers and or actual hardware may vary. Therefore the following considerations should be taken into account when applications use XFS form definition files with the purpose of running in a multi-vendor environment:

- Physical print area dimensions of printers are not identical
- Graphic printout may not be supported, which may limit the use of the FONT, CPI and LPI keywords
- Some printers may have a resolution of dots/mm rather than dots/inch, which may result in printouts with a specific CPI/LPI font resolution to be slightly off size
- Just-in-time form loading may not be supported by all Service Providers, which makes it impossible to create dynamic form files just before printing (which in return means that only the print data of the forms can be changed, not the -layout data such as the font and font size)
- Some form/media definition keywords may not be supported due to limitations of the hardware or software

3. New Info Commands

There are no new Info Commands.

4. Changes to Existing Info Commands

4.1. WFS_INF_PTR_STATUS

Description This command is used to request status information for the device.

Input Param None.

Output Param

```
LPWFSPTRSTATUS      lpStatus;

typedef struct _wfs_ptr_status
   {
   WORD                    fwDevice;
   WORD                    fwMedia;
   WORD                    fwPaper[WFS_PTR_SUPPLYSIZE];
   WORD                    fwToner;
   WORD                    fwInk;
   WORD                    fwLamp;
   ~~WORD                    fwRetractBin;~~
   ~~WORD                    usRetractCount;~~
   LPWFSPTRRETRACTBINS * lppRetractBins;
   USHORT               usMediaOnStacker;
   LPSTR                   lpszExtra;
   } WFSPTRSTATUS, * LPWFSPTRSTATUS;
```

fwDevice
Specifies the state of the print device as one of the following flags:

Value	Meaning
WFS_PTR_DEVONLINE	The device is online (i.e. powered on and operable).
WFS_PTR_DEVOFFLINE	The device is offline (e.g., the operator has taken the device offline by turning a switch or pulling out the device).
WFS_PTR_DEVPOWEROFF	The device is powered off or physically not connected.
WFS_PTR_DEVNODEVICE	There is no device intended to be there; e.g. this type of self service machine does not contain such a device or it is internally not configured.
WFS_PTR_DEVHWERROR	The device is inoperable due to a hardware error.
WFS_PTR_DEVUSERERROR	The device is present but a person is preventing proper device operation. The application should suspend the device from service until the service provider generates a device state change event indicating the condition of the device has changed, e.g., the error is removed (WFS_PTR_DEVONLINE) or a permanent error condition has occurred (WFS_PTR_DEVHWERROR).
WFS_PTR_DEVBUSY	The device is busy and unable to process an execute command at this time.

fwMedia
Specifies the state of the print media (i.e. receipt, statement, passbook, etc..) as one of the following values:

Value	Meaning
WFS_PTR_MEDIAPRESENT	Media is in the print position or on the stacker (i.e. a passbook in the parking position is not considered to be present).
WFS_PTR_MEDIANOTPRESENT	Media is not in the print position or on the stacker.
WFS_PTR_MEDIAJAMMED	Media is jammed in the device.
WFS_PTR_MEDIANOTSUPP	The capability to report the state of the print media is not supported by the device.
WFS_PTR_MEDIAUNKNOWN	The state of the print media cannot be determined with the device in its current state.
WFS_PTR_MEDIAENTERING	Media is at the entry/exit slot of the device.

fwPaper[…]
Specifies the state of the paper supplies. A number of paper supplies are defined below. Vendor specific paper supplies are defined starting from the end od the array. The maximum paper index is WFS_PTR_SUPPLYMAX

fwPaper[WFS_PTR_SUPPLYUPPER]
Specifies the state of the only paper supply or the upper paper supply, if more than one, as one of the following values:

Value	Meaning
WFS_PTR_PAPERFULL	The paper supply is full.
WFS_PTR_PAPERLOW	The paper supply is low.
WFS_PTR_PAPEROUT	The paper supply is empty.
WFS_PTR_PAPERNOTSUPP	Capability not supported by device.
WFS_PTR_PAPERUNKNOWN	Capability cannot be determined with device in its current state.

fwPaper[WFS_PTR_SUPPLYLOWER]
Specifies the state of the lower paper supply as one of the following values:

Value	Meaning
WFS_PTR_PAPERFULL	The paper supply is full.
WFS_PTR_PAPERLOW	The paper supply is low.
WFS_PTR_PAPEROUT	The paper supply is empty.
WFS_PTR_PAPERNOTSUPP	Capability not supported by device.
WFS_PTR_PAPERUNKNOWN	Capability cannot be determined with device in its current state.

fwPaper[WFS_PTR_SUPPLYEXTERNAL]
Specifies the state of the external paper supply as one of the following values:

Value	Meaning
WFS_PTR_PAPERFULL	The paper supply is full.
WFS_PTR_PAPERLOW	The paper supply is low.
WFS_PTR_PAPEROUT	The paper supply is empty.
WFS_PTR_PAPERNOTSUPP	Capability not supported by device.
WFS_PTR_PAPERUNKNOWN	Capability cannot be determined with device in its current state.

fwPaper[WFS_PTR_SUPPLYAUX]
Specifies the state of the auxiliary paper supply as one of the following values:

Value	Meaning
WFS_PTR_PAPERFULL	The paper supply is full.
WFS_PTR_PAPERLOW	The paper supply is low.
WFS_PTR_PAPEROUT	The paper supply is empty.
WFS_PTR_PAPERNOTSUPP	Capability not supported by device.
WFS_PTR_PAPERUNKNOWN	Capability cannot be determined with device in its current state.

fwPaper[WFS_PTR_SUPPLYAUX2]
Specifies the state of the second auxiliary paper supply as one of the following values:

Value	Meaning
WFS_PTR_PAPERFULL	The paper supply is full.
WFS_PTR_PAPERLOW	The paper supply is low.
WFS_PTR_PAPEROUT	The paper supply is empty.
WFS_PTR_PAPERNOTSUPP	Capability not supported by device.
WFS_PTR_PAPERUNKNOWN	Capability cannot be determined with device in its current state.

fwPaper[WFS_PTR_SUPPLYPARK]
Specifies the state of the parking station as one of the following values:

Value	Meaning
WFS_PTR_PAPERFULL	The parking station is busy.
WFS_PTR_PAPEROUT	The parking station is free.
WFS_PTR_PAPERNOTSUPP	Capability not supported by device.

WFS_PTR_PAPERUNKNOWN	Capability cannot be determined with device in its current state.

fwToner
Specifies the state of the toner or ink supply or the state of the ribbon as one of the following values:

Value	Meaning
WFS_PTR_TONERFULL	The toner or ink supply is full or the ribbon is OK.
WFS_PTR_TONERLOW	The toner or ink supply is low or the print contrast with a ribbon is weak.
WFS_PTR_TONEROUT	The toner or ink supply is empty or the print contrast with a ribbon is not sufficient any more.
WFS_PTR_TONERNOTSUPP	Capability not supported by device.
WFS_PTR_TONERUNKNOWN	Status of toner or ink supply or the ribbon cannot be determined with device in its current state.

fwInk
Specifies the status of the stamping ink in the printer as one of the following values:

Value	Meaning
WFS_PTR_INKFULL	Ink supply in device is full.
WFS_PTR_INKLOW	Ink supply in device is low.
WFS_PTR_INKOUT	Ink supply in device is empty.
WFS_PTR_INKNOTSUPP	Capability not supported by device.
WFS_PTR_INKUNKNOWN	Status of the stamping ink supply cannot be determined with device in its current state.

fwLamp
Specifies the status of the printer imaging lamp as one of the following values:

Value	Meaning
WFS_PTR_LAMPOK	The lamp is OK.
WFS_PTR_LAMPFADING	The lamp should be changed.
WFS_PTR_LAMPINOP	The lamp is inoperative.
WFS_PTR_LAMPNOTSUPP	Capability not supported by device.
WFS_PTR_LAMPUNKNOWN	Status of the imaging lamp cannot be determined with device in its current state.

lppRetractBins
Pointer to a NULL terminated array of pointers to WFSPTRRETRACTBINS structures (one for each supported bin). The first pointer holds the structure for bin one, the second for bin two and so on.

```
typedef struct _wfs_ptr_retract_bins
    {
    WORD        wRetractBin;
    USHORT      usRetractCount;
    } WFSPTRRETRACTBINS, * LPWFSPTRRETRACTBINS;
```

wRetractBin
Specifies the state of the printer retract bin as one of the following values.

Value	Meaning
WFS_PTR_RETRACTBINOK	The retract bin of the printer is in a healthy state.
WFS_PTR_RETRACTBINFULL	The retract bin of the printer is full.
WFS_PTR_RETRACTUNKNOWN	Status cannot be determined with device in its current state.
WFS_PTR_RETRACTBINHIGH	The retract bin of the printer is nearly full.

usRetractCount
The number of media retracted to this bin. This value is persistent: it may be reset to zero by the WFS_CMD_PTR_RESET_COUNT command.

usMediaOnStacker
The number of media on stacker; applicable only to printers with stacking capability.

lpszExtra
Pointer to a list of vendor-specific, or any other extended, information. The information is returned as a series of "*key=value*" strings so that it is easily extensible by service providers. Each string is null-terminated, with the final string terminating with two null characters.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Comments Applications which require or expect specific information to be present in the *lpszExtra* parameter may not be device or vendor-independent.

4.2. WFS_INF_PTR_CAPABILITIES

Description This command is used to request device capability information.

Input Param None.

Output Param LPWFSPTRCAPS lpCaps;

```
typedef struct _wfs_ptr_caps
   {
   WORD       wClass;
   WORD       fwType;
   BOOL       bCompound;
   WORD       wResolution;
   WORD       fwReadForm;
   WORD       fwWriteForm;
   WORD       fwExtents;
   WORD       fwControl;
   ~~USHORT     usMaxRetract;~~
   USHORT     usMaxMediaOnStacker;
   BOOL       bAcceptMedia;
   BOOL       bMultiPage;
   WORD       fwPaperSources;
   BOOL       bMediaTaken;
   USHORT     usRetractBins;
   LPUSHORT   lpusMaxRetract;
   WORD       fwImageType;
   WORD       fwFrontImageColorFormat;
   WORD       fwBackImageColorFormat;
   WORD       fwCodelineFormat;
   WORD       fwImageSource;
   WORD       fwCharSupport;
   BOOL       bDispensePaper;
   LPSTR      lpszExtra;
   } WFSPTRCAPS, * LPWFSPTRCAPS;
```

wClass
Specifies the logical service class, value is:
WFS_SERVICE_CLASS_PTR

fwType
Specifies the type(s) of the physical device driven by the logical service, as a combination of the following flags:

Value	Meaning
WFS_PTR_TYPERECEIPT	Device is a receipt printer.
WFS_PTR_TYPEPASSBOOK	Device is a passbook printer.
WFS_PTR_TYPEJOURNAL	Device is a journal printer.
WFS_PTR_TYPEDOCUMENT	Device is a document printer.
WFS_PTR_TYPESCANNER	Device is a scanner with printing capabilities.

bCompound
Specifies whether the logical device is part of a compound physical device and is either TRUE or FALSE.

wResolution
Specifies at which resolution(s) the physical device can print. Used by the application to select the level of print quality desired (e.g., as in Word for Windows); does not imply any absolute

level of resolution, only relative. Specified as a combination of the following flags:

Value	Meaning
WFS_PTR_RESLOW	Can print with low resolution.
WFS_PTR_RESMED	Can print with medium resolution.
WFS_PTR_RESHIGH	Can print with high resolution.
WFS_PTR_RESVERYHIGH	Can print with very high resolution.

fwReadForm
Specifies whether the device can read data from media, as a combination of the following flags:

Value	Meaning
WFS_PTR_READOCR	Device has OCR capability.
WFS_PTR_READMICR	Device has MICR capability.
WFS_PTR_READMSF	Device has MSF capability.
WFS_PTR_READBARCODE	Device has Barcode capability.
WFS_PTR_READPAGEMARK	Device has Page Mark capability.
WFS_PTR_READIMAGE	Device has imaging capability.
WFS_PTR_READEMPTYLINE	Device has capability to detect empty print lines for passbook printing.

fwWriteForm
Specifies whether the device can write data to the media, as a combination of the following flags (0 if none of the choices is supported):

Value	Meaning
WFS_PTR_WRITETEXT	Device has Text capability.
WFS_PTR_WRITEGRAPHICS	Device has Graphics capability.
WFS_PTR_WRITEOCR	Device has OCR capability.
WFS_PTR_WRITEMICR	Device has MICR capability.
WFS_PTR_WRITEMSF	Device has MSF capability.
WFS_PTR_WRITEBARCODE	Device has Barcode capability.
WFS_PTR_WRITESTAMP	Device has stamping capability.

fwExtents
Specifies whether the device is able to measure the inserted media, as a combination of the following flags (0 if none of the choices is supported):

Value	Meaning
WFS_PTR_EXTHORIZONTAL	Device has horizontal size detection capability.
WFS_PTR_EXTVERTICAL	Device has vertical size detection capability.

fwControl
Specifies the manner in which media can be controlled, as a combination of the following bit flags (0 if none of the choices is supported):

Value	Meaning
WFS_PTR_CTRLEJECT	Device can eject media.
WFS_PTR_CTRLPERFORATE	Device can perforate media.
WFS_PTR_CTRLCUT	Device can cut media.
WFS_PTR_CTRLSKIP	Device can skip to mark.
WFS_PTR_CTRLFLUSH	Device can be sent data that is buffered internally, and flushed to the printer on request.
WFS_PTR_CTRLRETRACT	Device can retract media
WFS_PTR_CTRLSTACK	Device can stack media items before ejecting as a bundle.
WFS_PTR_CTRLPARTIALCUT	Device can partially cut the media.
WFS_PTR_CTRLALARM	Device can ring a bell, beep or otherwise sound an audible alarm.
WFS_PTR_CTRLATPFORWARD	Capability to turn one page forward.
WFS_PTR_CTRLATPBACKWARD	Capability to turn one page backward.
WFS_PTR_CTRLTURNMEDIA	Device can turn inserted media.
WFS_PTR_CTRLSTAMP	Device can stamp on media.
WFS_PTR_CTRLPARK	Device can park a document into the parking station.

usMaxMediaOnStacker
Specifies the maximum number of media items that the stacker can hold (zero if not available).

bAcceptMedia
Specifies whether the device is able to accept media while no execute command is running that is waiting explicitly for media to be inserted. Its value is either TRUE or FALSE.

bMultiPage
Specifies whether the device is able to support multiple page print jobs. Its value is either TRUE or FALSE.

fwPaperSources
Specifies the Paper sources available for this printer as a combination of the following bit flags:

Value	Meaning
WFS_PTR_PAPERUPPER	Indicates an upper paper source is available, devices with only one paper supply must indicate WFS_PTR_PAPERUPPER as being available.
WFS_PTR_PAPERLOWER	Indicates a lower paper source is available.
WFS_PTR_PAPEREXTERNAL	Indicates an external paper source (such as envelope tray or single sheet feed) is available.
WFS_PTR_PAPERAUX	An auxiliary paper source is available.

bMediaTaken
Specifies whether the device is able to detect when the media is taken from the exit slot. If FALSE, the WFS_SRVE_PTR_MEDIATAKEN event is not fired. Its value is either TRUE or FALSE.

usRetractBins
Specifies the number of retract bins (zero if not supported).

lpusMaxRetract
Pointer to an array of the length *usRetractBins* with the maximum number of media items that each retract bin can hold (one count for each supported bin, starting from 0 for bin number one to *usRetractBins*-1 for bin number *usRetractBins*). NULL pointer if the device has no retract bin.

fwImageType
Specifies the image format supported by this device, as a combination of following flags (0 if not supported):

Value	Meaning
WFS_PTR_IMAGETIF	The device can return scanned images in TIFF 6.0 format.
WFS_PTR_IMAGEWMF	The device can return scanned images in WMF (Windows Metafile) format.
WFS_PTR_IMAGEBMP	The device can return scanned images in windows BMP format.

fwFrontImageColorFormat
Specifies the front image color formats supported by this device, as a combination of following flags (0 if not supported):

Value	Meaning
WFS_PTR_IMAGECOLORBINARY	The device can return scanned images in binary (image contains two colors, usually the colors back and white).
WFS_PTR_IMAGECOLORGRAYSCALE	The device can return scanned images in gray scale (image contains multiple gray colors).
WFS_PTR_IMAGECOLORFULL	The device can return scanned images in full color (image contains colors like red, green, blue etc.).

fwBackImageColorFormat
Specifies the back image color formats supported by this device, as a combination of following flags (0 if not supported):

Value	Meaning
WFS_PTR_IMAGECOLORBINARY	The device can return scanned images in binary (image contains two colors, usually the colors back and white).

WFS_PTR_IMAGECOLORGRAYSCALE	The device can return scanned images in gray scale (image contains multiple gray colors).
WFS_PTR_IMAGECOLORFULL	The device can return scanned images in full color (image contains colors like red, green, blue etc.).

fwCodelineFormat
Specifies the code line (MICR data) formats supported by this device, as a combination of following flags (0 if not supported):

Value	Meaning
WFS_PTR_CODELINECMC7	The device can read CMC7 code lines.
WFS_PTR_CODELINEE13B	The device can read E13B code lines.
WFS_PTR_CODELINEOCR	The device can read code lines using Optical Character Recognition.

fwImageSource
Specifies the source for the read image command supported by this device, as a combination of the following flags (0 if not supported):

Value	Meaning
WFS_PTR_IMAGEFRONT	The device can scan the front image of the document.
WFS_PTR_IMAGEBACK	The device can scan the back image of the document.
WFS_PTR_CODELINE	The device can recognize the code line.

fwCharSupport
One or more flags specifying the character sets, in addition to single byte ASCII, that is supported by the service provider:

Value	Meaning
WFS_PTR_ASCII	ASCII is supported for XFS forms.
WFS_PTR_UNICODE	UNICODE is supported for XFS forms.

For *fwCharSupport*, a service provider can support ONLY ASCII forms or can support BOTH ASCII and UNICODE forms. A service provider can not support UNICODE forms without also supporting ASCII forms.

bDispensePaper
Specifies whether the device is able to dispense paper. Its value is either TRUE or FALSE.

lpszExtra
Points to a list of vendor-specific, or any other extended, information. The information is returned as a series of "*key=value*" strings so that it is easily extensible by service providers. Each string is null-terminated, with the final string terminating with two null characters.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Comments Applications which require or expect specific information to be present in the *lpszExtra* parameter may not be device or vendor-independent.

4.3. WFS_INF_PTR_QUERY_FORM

Description This command is used to retrieve details of the definition of a specified form.

Input Param
```
LPSTR        lpszFormName;
```

lpszFormName
Points to the null-terminated form name on which to retrieve details.

Output Param `LPWFSFRMHEADER      lpHeader;`

```
typedef struct _wfs_frm_header
   {
   LPSTR     lpszFormName;
   WORD      wBase;
   WORD      wUnitX;
   WORD      wUnitY;
   WORD      wWidth;
   WORD      wHeight;
   WORD      wAlignment;
   WORD      wOrientation;
   WORD      wOffsetX;
   WORD      wOffsetY;
   WORD      wVersionMajor;
   WORD      wVersionMinor;
   LPSTR     lpszUserPrompt;
   WORD      fwCharSupport;
   LPSTR     lpszFields;
   } WFSFRMHEADER, * LPWFSFRMHEADER;
```

lpszFormName
Specifies the null-terminated name of the form.

wBase
Specifies the base unit of measurement of the form and can be one of the following values:

Value	Meaning
WFS_FRM_INCH	The base unit is inches.
WFS_FRM_MM	The base unit is millimeters.
WFS_FRM_ROWCOLUMN	The base unit is rows and columns.

wUnitX
Specifies the horizontal resolution of the base units as a fraction of the *wBase* value. For example, a value of 16 applied to the base unit WFS_FRM_INCH means that the base horizontal resolution is 1/16".

wUnitY
Specifies the vertical resolution of the base units as a fraction of the *wBase* value. For example, a value of 10 applied to the base unit WFS_FRM_MM means that the base vertical resolution is 0.1 mm.

wWidth
Specifies the width of the form in terms of the base horizontal resolution.

wHeight
Specifies the height of the form in terms of the base vertical resolution.

wAlignment
Specifies the relative alignment of the form on the media and can be one of the following values:

Value	Meaning
WFS_FRM_TOPLEFT	The form is aligned relative to the top and left edges of the media.
WFS_FRM_TOPRIGHT	The form is aligned relative to the top and right edges of the media.
WFS_FRM_BOTTOMLEFT	The form is aligned relative to the bottom and left edges of the media.
WFS_FRM_BOTTOMRIGHT	The form is aligned relative to the bottom and right edges of the media.

wOrientation
Specifies the orientation of the form and can be one of the following values:

Value	Meaning
WFS_FRM_PORTRAIT	The orientation of the form is portrait.
WFS_FRM_LANDSCAPE	The orientation of the form is landscape.

wOffsetX
Specifies the horizontal offset of the position of the top-left corner of the form, relative to the left or right edge specified by *wAlignment*. This value is specified in terms of the base

horizontal resolution and is always positive.

wOffsetY
Specifies the vertical offset of the position of the top-left corner of the form, relative to the top or bottom edge specified by *wAlignment*. This value is specified in terms of the base vertical resolution and is always positive.

wVersionMajor
Specifies the major version of the form.

wVersionMinor
Specifies the minor version of the form.

lpszUserPrompt
Pointer to a null-terminated user prompt string.

fwCharSupport
A single flag specifying the Character Set in which the form is encoded:

Value	Meaning
WFS_PTR_ASCII	ASCII is supported for XFS forms initial data values and FORMAT strings.
WFS_PTR_UNICODE	UNICODE is supported for XFS forms initial data values and FORMAT strings.

lpszFields
Pointer to a list of null-terminated field names, with the final name terminating with two null characters.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_FORMNOTFOUND	The specified form cannot be found.
WFS_ERR_PTR_FORMINVALID	The specified form is invalid.

Comments None.

4.4. WFS_INF_PTR_QUERY_MEDIA

Description This command is used to retrieve details of the definition of a specified media.

Input Param `LPSTR         lpszMediaName;`

lpszMediaName
Pointer to the null-terminated media name about which to retrieve details.

Output Param `LPWFSFRMMEDIA lpMedia;`

```
typedef struct _wfs_frm_media
    {
    WORD       fwMediaType;
    WORD       wBase;
    WORD       wUnitX;
    WORD       wUnitY;
    WORD       wSizeWidth;
    WORD       wSizeHeight;
    WORD       wPageCount;
    WORD       wLineCount;
    WORD       wPrintAreaX;
    WORD       wPrintAreaY;
    WORD       wPrintAreaWidth;
    WORD       wPrintAreaHeight;
    WORD       wRestrictedAreaX;
    WORD       wRestrictedAreaY;
    WORD       wRestrictedAreaWidth;
    WORD       wRestrictedAreaHeight;
    WORD       wStagger;
    WORD       wFoldType;
    WORD       wPaperSources;
    } WFSFRMMEDIA, * LPWFSFRMMEDIA;
```

fwMediaType
Specifies the type of media as one of the following values:

Value	Meaning
WFS_FRM_MEDIAGENERIC	The media is a generic media, i.e. a single sheet.
WFS_FRM_MEDIAPASSBOOK	The media is a passbook media.
WFS_FRM_MEDIAMULTIPART	The media is a multipart media.

wBase
Specifies the base unit of measurement of the form and can be one of the following values:

Value	Meaning
WFS_FRM_INCH	The base unit is inches.
WFS_FRM_MM	The base unit is millimeters.
WFS_FRM_ROWCOLUMN	The base unit is rows and columns.

wUnitX
Specifies the horizontal resolution of the base units as a fraction of the *wBase* value. For example, a value of 16 applied to the base unit WFS_FRM_INCH means that the base horizontal resolution is 1/16".

wUnitY
Specifies the vertical resolution of the base units as a fraction of the *wBase* value. For example, a value of 10 applied to the base unit WFS_FRM_MM means that the base vertical resolution is 0.1 mm.

wSizeWidth
Specifies the width of the media in terms of the base horizontal resolution.

wSizeHeight
Specifies the height of the media in terms of the base vertical resolution.

wPageCount
Specifies the number of pages in a media of type WFS_FRM_MEDIAPASSBOOK.

wLineCount
Specifies the number of lines on a page for a media of type WFS_FRM_MEDIAPASSBOOK.

wPrintAreaX
Specifies the horizontal offset of the printable area relative to the top left corner of the media in terms of the base horizontal resolution.

wPrintAreaY
Specifies the vertical offset of the printable area relative to the top left corner of the media in terms of the base vertical resolution.

wPrintAreaWidth
Specifies the printable area width of the media in terms of the base horizontal resolution.

wPrintAreaHeight
Specifies the printable area height of the media in terms of the base vertical resolution.

wRestrictedAreaX
Specifies the horizontal offset of the restricted area relative to the top left corner of the media in terms of the base horizontal resolution.

wRestrictedAreaY
Specifies the vertical offset of the restricted area relative to the top left corner of the media in terms of the base vertical resolution.

wRestrictedAreaWidth
Specifies the restricted area width of the media in terms of the base horizontal resolution.

wRestrictedAreaHeight
Specifies the restricted area height of the media in terms of the base vertical resolution.

w*Stagger*
Specifies the staggering from the top in terms of the base vertical resolution for a media of type WFS_FRM_MEDIAPASSBOOK.

wFoldType
Specified the type of fold (vertical, horizontal or none) for a media of type WFS_FRM_MEDIAPASSBOOK as one of the following values:

Value	Meaning
WFS_FRM_FOLDNONE	Passbook has no fold.
WFS_FRM_FOLDHORIZONTAL	Passbook has a horizontal fold.
WFS_FRM_FOLDVERTICAL	Passbook has a vertical fold.

wPaperSources
Specifies the Paper sources to use when printing forms using this media as one of the following flags

Value	Meaning
WFS_PTR_PAPERUPPER	Use the only or the upper paper source.
WFS_PTR_PAPERLOWER	Use the lower paper source.
WFS_PTR_PAPEREXTERNAL	Use the external paper source.
WFS_PTR_PAPERAUX	Use the auxiliary paper source.
WFS_PTR_PAPERAUX2	Use the second auxiliary paper source.
WFS_PTR_PAPERPARK	Use the parking station.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_MEDIANOTFOUND	The specified media definition cannot be found.
WFS_ERR_PTR_MEDIAINVALID	The specified media definition is invalid.

Comments None.

4.5. WFS_INF_PTR_QUERY_FIELD

Description This command is used to retrieve details of the definition of a single or all fields on a specified form.

Input Param `LPWFSPTRQUERYFIELD lpQueryField;`

```
typedef struct _wfs_ptr_query_field
    {
    LPSTR            lpszFormName;
    LPSTR            lpszFieldName;
    } WFSPTRQUERYFIELD, * LPWFSPTRQUERYFIELD;
```

lpszFormName
Pointer to the null-terminated form name.

lpszFieldName
Pointer to the null-terminated name of the field about which to retrieve details.
If the value of *lpszFieldName* is NULL, then details are retrieved for all fields on the form.
Depending upon whether the form is encoded in UNICODE representation either the *lpszInitialValue* or *lpszUNICODEInitialValue* output fields are used to retrieve the field Initial Value.

Output Param `LPWFSFRMFIELD *     lppFields;`

lppFields
Pointer to a null-terminated array of pointers to field definition structures:

```
  typedef struct _wfs_frm_field
      {
      LPSTR     lpszFieldName;
      WORD      wIndexCount;
      WORD      fwType;
      WORD      fwClass;
      WORD      fwAccess;
      WORD      fwOverflow;
      LPSTR     lpszInitialValue;
      LPWSTR    lpszUNICODEInitialValue;
      LPSTR     lpszFormat;
      LPWSTR    lpszUNICODEFormat;
      } WFSFRMFIELD, * LPWFSFRMFIELD;
```

lpszFieldName
Pointer to the null-terminated field name.

wIndexCount
Specifies the number of entries for an index field. A value of zero indicates that this field is not an index field. Index fields are typically used to present information in a tabular fashion.

fwType
Specifies the type of field and can be one of the following:

Value	Meaning
WFS_FRM_FIELDTEXT	The field is a text field.
WFS_FRM_FIELDMICR	The field is a Magnetic Ink Character Recognition field.
WFS_FRM_FIELDOCR	The field is an Optical Character Recognition field.
WFS_FRM_FIELDMSF	The field is a Magnetic Stripe Facility field.
WFS_FRM_FIELDBARCODE	The field is a Barcode field.
WFS_FRM_FIELDGRAPHIC	The field is a Graphic field
WFS_FRM_FIELDPAGEMARK	The field is a Page Mark field

fwClass
Specifies the class of the field and can be one of the following values:

Value	Meaning
WFS_FRM_CLASSSTATIC	The field data cannot be set by the application.
WFS_FRM_CLASSOPTIONAL	The field data can be set by the application.
WFS_FRM_CLASSREQUIRED	The field data must be set by the application.

fwAccess
Specifies whether the field is to be used for input, output, or both and can be a combination of the following ~~bit~~ flags:

Value	Meaning
WFS_FRM_ACCESSREAD	The field is used for input.
WFS_FRM_ACCESSWRITE	The field is used for ouput.

fwOverflow
Specifies how an overflow of field data should be handled and can be one of the following values:

Value	Meaning
WFS_FRM_OVFTERMINATE	Return an error and terminate printing of the form.
WFS_FRM_OVFTRUNCATE	Truncate the field data to fit in the field.
WFS_FRM_OVFBESTFIT	Fit the text in the field.
WFS_FRM_OVFOVERWRITE	Print the field data beyond the extents of the field boundary.
WFS_FRM_OVFWORDWRAP	If the field can hold more than one line the text is wrapped around.

lpszInitialValue
The initial value of the field. When the form is printed (using WFS_CMD_PTR_PRINT_FORM), this value will be used if another value is not provided. This value can be NULL if the parameter is not specified in the field definition or the form is encoded in UNICODE.

lpszUNICODEInitialValue
The initial value of the field when form is encoded in UNICODE. When the form is printed (using WFS_CMD_PTR_PRINT_FORM), this value will be used if another value is not provided. This value can be NULL if the parameter is not specified in the field definition or the form is not encoded in UNICODE.

lpszFormat
Format string as defined in the form for this field. This value can be NULL if the parameter is not specified in the field definition or the form is encoded in UNICODE.

lpszUNICODEFormat
Format string as defined in the form for this field when form is encoded in UNICODE. This value can be NULL if the parameter is not specified in the field definition or the form is not encoded in UNICODE.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_FORMNOTFOUND	The specified form cannot be found.
WFS_ERR_PTR_FIELDNOTFOUND	The specified field cannot be found.
WFS_ERR_PTR_FORMINVALID	The specified form is invalid.
WFS_ERR_PTR_FIELDINVALID	The specified field is invalid.

Comments None.

5. New Execute Commands

5.1. *WFS_CMD_PTR_RESET*

Description This command is used by the application to perform a hardware reset which will attempt to return the PTR device to a known good state. This command does not over-ride a lock obtained on another application or service handle.

The device will attempt to retract or eject any items found anywhere within the device. This may not always be possible because of hardware problems. The WFS_SRVE_PTR_MEDIADETECTED event will inform the application where items were actually moved to.

Input Param `LPWFSPTRRESET       lpReset;`

Specifies where media should be moved to that is found in the device. If the application does not wish to specify a position it can set this value to NULL. In this case the service provider will determine where to move any items found.

```
typedef struct _wfs_ptr_reset
   {
   DWORD     dwMediaControl;
   USHORT    usRetractBinNumber;
   } WFSPTRRESET, * LPWFSPTRRESET;
```

dwMediaControl
Pointer to a value which specifies the manner in which the media should be handled, as a combination of the following bit-flags:

Value	Meaning
WFS_PTR_CTRLEJECT	Eject the media.
WFS_PTR_CTRLRETRACT	Retract the media to retract bin number one.

usRetractBinNumber
Number of the retract bin the media is retracted to. This number has to be between one and the number of bins supported by this device. It is only relevant if *dwMediaControl* equals WFS_PTR_CTRLRETRACT.

Output Param None.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_SHUTTERFAIL	Open or close of the shutter failed due to manipulation or hardware error.
WFS_ERR_PTR_RETRACTBINFULL	The retain bin is full; no more media can be retained. The current media is still in the device.
WFS_ERR_PTR_MEDIAJAMMED	The media is jammed. Operator intervention is required.
WFS_ERR_PTR_PAPERJAMMED	The paper is jammed.

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_SRVE_PTR_MEDIADETECTED	A media is detected in the device during a reset operation.

Comments This command is used by an application control program to cause a device to reset itself to a known good condition.

5.2. WFS_CMD_PTR_RETRACT_MEDIA

Description The media is removed from its present position (media inserted into device, media entering, unknown position) and stored in one of the retract bins. An event is sent if the storage capacity of the specified retract bin is reached. If the bin is already full and the command cannot be executed, an error is returned and the media remains in its present position.

Input Param
```
LPUSHORT            lpusBinNumber;
```

lpusBinNumber
Pointer to the number of one of the retract bins. This number has to be between one and the number of bins supported by this device.

Output Param
```
LPUSHORT            lpusBinNumber;
```

lpusBinNumber
Pointer to the number of the retract bin where the media has actually been deposited.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_NOMEDIAPRESENT	No media present on retract. Either there was no media present when the command was called or the media was removed during the retract.
WFS_ERR_PTR_RETRACTBINFULL	The retain bin is full; no more media can be retained. The current media is still in the device.
WFS_ERR_PTR_MEDIAJAMMED	The media is jammed. Operator intervention is required.

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_USRE_PTR_RETRACTBINTHRESHOLD	The retract bin is full; operator intervention is required. Note that this event is sent only once, at the point at which the bin becomes full. It is sent with WFS_PTR_RETRACTBINFULL or WFS_PTR_RETRACTBINHIGH status.

Comments If a retain request is received by a device with no retract capability, the WFS_ERR_UNSUPP_COMMAND error is returned.

5.3. WFS_CMD_PTR_DISPENSE_PAPER

Description This command is used to move paper (which can also be a new passbook) from a paper source into the print position.

Input Param `WORD   wPaperSource;`

wPaperSource
Specifies the Paper source to dispense from. Possible values are:

Value	Meaning
WFS_PTR_PAPERANY	Any paper source can be used, it is determined by the service.
WFS_PTR_PAPERUPPER	Use the only paper source or the upper paper source, if there is more than one paper supply.
WFS_PTR_PAPERLOWER	Use the lower paper source.
WFS_PTR_PAPEREXTERNAL	Use the external paper.
WFS_PTR_PAPERAUX	Use the auxiliary paper source.
WFS_PTR_PAPERAUX2	Use the second auxiliary paper source.
WFS_PTR_PAPERPARK	Use the parking station paper source.

Output Param None.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_PAPERJAMMED	The paper is jammed.
WFS_ERR_PTR_PAPEROUT	The paper supply is empty.
WFS_ERR_PTR_SEQUENCEINVALID	Programming error. Invalid command sequence (e.g. there is already media in the print position).

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_USRE_PTR_PAPERTHRESHOLD	The paper supply is low or empty, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_PAPERLOW or WFS_PTR_PAPEROUT status.

Comments None.

6. Changes to existing Execute Commands

6.1. WFS_CMD_PTR_CONTROL_MEDIA

Description This command is used to control a form drawn in by the device (e.g. after reading or in case of termination of an application request).

If an eject operation is specified, it completes when the media is moved to the exit slot. A service event is generated when the media has been taken by the user (only if field bMediaTaken defined in structure WFSPTRCAPS is equal to TRUE).

Input Param `LPDWORD lpdwMediaControl;`

lpdwMediaControl
Pointer to a value which specifies the manner in which the media should be handled, as a combination of the following bit-flags:

Value	Meaning
WFS_PTR_CTRLEJECT	Flush any data to the printer that has not yet been printed from previous WFS_CMD_PTR_PRINT_FORM commands, then eject the media.
WFS_PTR_CTRLPERFORATE	Flush data as above, then perforate the media.
WFS_PTR_CTRLCUT	Flush data as above, then cut the media. For printers which have the ability to stack multiple cut sheets and deliver them as a single bundle to the customer, cut causes the media to be stacked and eject causes the bundle to be moved to the exit slot.
WFS_PTR_CTRLSKIP	Flush data as above, then skip the media to mark.
WFS_PTR_CTRLFLUSH	Flush any data to the printer that has not yet been printed from previous WFS_CMD_PTR_PRINT_FORM commands.
WFS_PTR_CTRLRETRACT	Flush data as above, then retract the media to retract bin number one, for devices with more than one bin the command WFS_CMD_PTR_RETRACT_MEDIA should be used if the media should be retracted to another bin than bin number one.
WFS_PTR_CTRLSTACK	Flush data as above, then move the media item on the internal stacker.
WFS_PTR_CTRLPARTIALCUT	Flush the data as above, then partially cut the media.
WFS_PTR_CTRLALARM	Caused the printer to ring a bell, beep, or otherwise sound an audible alarm.
WFS_PTR_CTRLATPFORWARD	Flush the data as above, then turn one page forward.
WFS_PTR_CTRLATPBACKWARD	Flush the data as above, then turn one page backward.
WFS_PTR_CTRLTURNMEDIA	Flush the data as above, then turn inserted media.
WFS_PTR_CTRLSTAMP	Flush the data as above, then stamp on inserted media.
WFS_PTR_CTRLPARK	Park the media in the parking station.

It is not possible to combine the flags WFS_PTR_CTRLEJECT, WFS_PTR_CTRLRETRACT and WFS_PTR_CTRLPARK with each other. In this case the command completes with WFS_ERR_INVALID_DATA.

An application should be aware that the sequence of the actions is not guaranteed if more than one flag is specified in this parameter.

Output Param None.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_NOMEDIAPRESENT	No media is present in the device.
WFS_ERR_PTR_FLUSHFAIL	The device was not able to flush data.
WFS_ERR_PTR_RETRACTBINFULL	The retract bin is full. No more media can be retracted. The current media is still in the device.
WFS_ERR_PTR_STACKERFULL	The internal stacker is full. No more media can be moved to the stacker.
WFS_ERR_PTR_PAGETURNFAIL	The device was not able to turn the page.
WFS_ERR_PTR_MEDIATURNFAIL	The device was not able to turn the inserted media.
WFS_ERR_PTR_SHUTTERFAIL	Open or close of the shutter failed due to manipulation or hardware error.
WFS_ERR_PTR_MEDIAJAMMED	The media is jammed. Operator intervention is required.
WFS_ERR_PTR_PAPERJAMMED	The paper is jammed.
WFS_ERR_PTR_PAPEROUT	The paper supply is empty.
WFS_ERR_PTR_INKOUT	No stamping possible, stamping ink supply empty.
WFS_ERR_PTR_TONEROUT	Toner or ink supply is empty or printing contrast with ribbon is not sufficient.
WFS_ERR_PTR_SEQUENCEINVALID	Programming error. Invalid command sequence (e.g. WFS_PTR_CTRLPARK and the parking station is busy).

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_USRE_PTR_RETRACTBINTHRESHOLD	The retract bin is high or full; operator intervention is required. Note that this event is sent only once, at the point at which the bin becomes high or full. It is sent with WFS_PTR_RETRACTBINHIGH or WFS_PTR_RETRACTBINFULL status.
WFS_SRVE_PTR_MEDIATAKEN	The media has been taken by the user.
WFS_USRE_PTR_PAPERTHRESHOLD	The paper supply is low or empty, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_PAPERLOW or WFS_PTR_PAPEROUT status.
WFS_USRE_PTR_TONERTHRESHOLD	The toner or ink supply is low or empty or the printing contrast with ribbon is weak or not sufficient, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_TONERLOW or WFS_PTR_TONEROUT status.
WFS_USRE_PTR_INKTHRESHOLD	The stamping ink supply is low or empty, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_INKLOW or WFS_PTR_INKOUT status.

Comments None.

6.2. WFS_CMD_PTR_PRINT_FORM

Description This command is used to print a form by merging the supplied variable field data with the defined form and field data specified in the form. If no media is present, the device waits for the period of time specified by the *dwTimeOut* parameter in the **WFSExecute** call for media to be inserted from the external paper source.

Input Param

```
LPWFSPTRPRINTFORM   lpPrintForm;

typedef struct _wfs_ptr_print_form
   {
   LPSTR      lpszFormName;
   LPSTR      lpszMediaName;
   WORD       wAlignment;
   WORD       wOffsetX;
   WORD       wOffsetY;
   WORD       wResolution;
   DWORD      dwMediaControl;
   LPSTR      lpszFields;
   LPWSTR     lpszUNICODEFields;
   WORD       wPaperSource;
   } WFSPTRPRINTFORM, * LPWFSPTRPRINTFORM;
```

lpszFormName
Pointer to the null-terminated form name.

lpszMediaName
Pointer to the null-terminated media name.

wAlignment
Specifies the alignment of the form on the physical medium, as one of the following values:

Value	Meaning
WFS_PTR_ALNUSEFORMDEFN	Use the alignment specified in the form definition.
WFS_PTR_ALNTOPLEFT	Align form to top left of physical medium.
WFS_PTR_ALNTOPRIGHT	Align form to top right of physical medium.
WFS_PTR_ALNBOTTOMLEFT	Align form to bottom left of physical medium.
WFS_PTR_ALNBOTTOMRIGHT	Align form to bottom right of physical medium.

wOffsetX
Specifies the horizontal offset of the form, relative to the horizontal alignment specified in *wAlignment*, in horizontal resolution units (from form definition); always a positive number (i.e., if aligned to the right side of the medium, means offset the form to the left). A value of WFS_PTR_OFFSETUSEFORMDEFN indicates that the *xoffset* value from the form definition should be used.

wOffsetY
Specifies the vertical offset of the form, relative to the vertical alignment specified in *wAlignment*, in vertical resolution units (from form definition); always a positive number (i.e., if aligned to the bottom of the medium, means offset the form upward). A value of WFS_PTR_OFFSETUSEFORMDEFN indicates that the *yoffset* value from the form definition should be used.

wResolution
Specifies the resolution in which to print the form. Possible values are:

Value	Meaning
WFS_PTR_RESLOW	Print form with low resolution.
WFS_PTR_RESMED	Print form with medium resolution.
WFS_PTR_RESHIGH	Print form with high resolution.
WFS_PTR_RESVERYHIGH	Print form with very high resolution.

dwMediaControl
Specifies the manner in which the media should be handled after the printing was done, as a combination of the flags described under WFS_CMD_PTR_CONTROL_MEDIA. A zero value of this parameter means to do none of these actions, as when printing multiple forms on a single page.

lpszFields
Pointer to a series of "<FieldName>=<FieldValue>" strings, where each string is null-terminated with the entire field string terminating with two null characters. If the field is an index field, then the syntax of the string is instead "<FieldName>[<index>]=<FieldValue>", where <index> specifies the zero-based element of the index field.

lpszUNICODEFields
Pointer to a series of "<FieldName>=<FieldValue>" UNICODE strings, where each string is null-terminated with the entire field string terminating with two null characters. If the field is an index field, then the syntax of the string is instead "<FieldName>[<index>]=<FieldValue>", where <index> specifies the zero-based element of the index field.

The *lpszUNICODEFields* field should only be used if the form is encoded in UNICODE representation. This can be determined with the WFS_PTR_INF_QUERY_FORM command.

wPaperSource
Specifies the Paper source to use when printing this form. When the value is zero, then the paper source is determined from the media definition. This parameter is ignored if there is already paper in the print position. Possible values are:

Value	Meaning
WFS_PTR_PAPERANY	Any paper source can be used, it is determined by the service.
WFS_PTR_PAPERUPPER	Use the only paper source or the upper paper source, if there is more than one paper supply.
WFS_PTR_PAPERLOWER	Use the lower paper source.
WFS_PTR_PAPEREXTERNAL	Use the external paper source (such as envelope tray or single sheet feed).
WFS_PTR_PAPERAUX	Use the auxiliary paper source.
WFS_PTR_PAPERAUX2	Use the second auxiliary paper source.
WFS_PTR_PAPERPARK	Use the parking station.

Output Param None.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_FORMNOTFOUND	The specified form definition cannot be found.
WFS_ERR_PTR_FLUSHFAIL	The device was not able to flush data.
WFS_ERR_PTR_MEDIAOVERFLOW	The form overflowed the media.
WFS_ERR_PTR_FIELDSPECFAILURE	The syntax of the *lpszFields* member is invalid.
WFS_ERR_PTR_FIELDERROR	An error occurred while processing a field, causing termination of the print request. An execute event WFS_EXEE_PTR_FIELDERROR is posted with the details.
WFS_ERR_PTR_MEDIANOTFOUND	The specified media definition cannot be found.
WFS_ERR_PTR_MEDIAINVALID	The specified media definition is invalid.
WFS_ERR_PTR_FORMINVALID	The specified form definition is invalid.
WFS_ERR_PTR_MEDIASKEWED	The media skew exceeded the limit in the form definition.
WFS_ERR_PTR_RETRACTBINFULL	The retract bin is full. No more media can be retracted. The current media is still in the device.
WFS_ERR_PTR_STACKERFULL	The internal stacker is full. No more media can be moved to the stacker.
WFS_ERR_PTR_PAGETURNFAIL	The device was not able to turn the page.
WFS_ERR_PTR_MEDIATURNFAIL	The device was not able to turn the inserted media.
WFS_ERR_PTR_SHUTTERFAIL	Open or close of the shutter failed due to manipulation or hardware error.
WFS_ERR_PTR_MEDIAJAMMED	The media is jammed. Operator intervention is required.
WFS_ERR_PTR_CHARSETDATA	Character set(s) supported by service provider is inconsistent with use of *lpszFields* or *lpszUNICODEFields* fields.

WFS_ERR_PTR_PAPERJAMMED	The paper is jammed.
WFS_ERR_PTR_PAPEROUT	The paper supply is empty.
WFS_ERR_PTR_INKOUT	No stamping possible, stamping ink supply empty.
WFS_ERR_PTR_TONEROUT	Toner or ink supply is empty or printing contrast with ribbon is not sufficient.
WFS_ERR_PTR_SEQUENCEINVALID	Programming error. Invalid command sequence (e.g. *dwMediaControl* = WFS_PTR_CTRLPARK and park position is busy).

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_EXEE_PTR_NOMEDIA	No media is present in the device.
WFS_EXEE_PTR_MEDIAINSERTED	Media has been inserted into the device.
WFS_EXEE_PTR_FIELDERROR	A fatal error occurred while processing a field.
WFS_EXEE_PTR_FIELDWARNING	A non-fatal error occurred while processing a field.
WFS_USRE_PTR_RETRACTBINTHRESHOLD	The retract bin is full; operator intervention is required. Note that this event is sent only once, at the point at which the bin becomes full. It is sent with WFS_PTR_RETRACTBINFULL or WFS_PTR_RETRACTBINHIGH status.
WFS_SRVE_PTR_MEDIATAKEN	The media has been taken by the user.
WFS_USRE_PTR_PAPERTHRESHOLD	The paper supply is low or empty, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_PAPERLOW or WFS_PTR_PAPEROUT status.
WFS_USRE_PTR_TONERTHRESHOLD	The toner or ink supply is low or empty or the printing contrast with ribbon is weak or not sufficient, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_TONERLOW or WFS_PTR_TONEROUT status.
WFS_USRE_PTR_INKTHRESHOLD	The stamping ink supply is low or empty, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_INKLOW or WFS_PTR_INKOUT status.

Comments All error codes (except WFS_ERR_PTR_NOMEDIAPRESENT) and events listed under the WFS_CMD_PTR_CONTROL_MEDIA command description can also occur on this command.

An invalid field name is treated as a WFS_EXEE_PTR_FIELDWARNING event with WFS_PTR_FIELDNOTFOUND status. A WFS_EXEE_PTR_FIELDWARNING event is returned with WFS_PTR_FIELDOVERFLOW status if the data overflows the field, and the field definition OVERFLOW value is TRUNCATE, BESTFIT, OVERWRITE or WORDWRAP. Other field-related problems generate a field error return and event.

The application will use *lpszFields* or *lpszUNICODEFields* as an input parameter, depending upon the service provider capabilities. Legacy (non-UNICODE aware) applications will only use the *lpszFields* field. UNICODE applications can use either the *lpszFields* or *lpszUNICODEFields* fields, provided the service provider is UNICODE compliant.

6.3. WFS_CMD_PTR_READ_FORM

Description This command is used to read data from input fields on the specified form. These input fields may consist of MICR, OCR, MSF, BARCODE, or PAGEMARK input fields. These input fields may also consist of TEXT fields for purposes of detecting available passbook print lines with passbook printers supporting such capability. If no media is present, the device waits for the period of time specified by the *dwTimeOut* parameter in the **WFSExecute** call for media to be inserted.

Input Param

```
LPWFSPTRREADFORM    lpReadForm;

typedef struct _wfs_ptr_read_form
   {
   LPSTR     lpszFormName;
   LPSTR     lpszFieldNames;
   LPSTR     lpszMediaName;
   DWORD     dwMediaControl;
   } WFSPTRREADFORM, * LPWFSPTRREADFORM;
```

lpszFormName
Pointer to the null-terminated name of the form.

lpszFieldNames
Pointer to a list of null-terminated field names from which to read input data, with the final name terminating with two null characters. If this value is NULL, then read data from all input fields on the form.

lpszMediaName
Pointer to the null-terminated media name.

dwMediaControl
Specifies the manner in which the media should be handled after the reading was done and can be a combination of the flags described under WFS_CMD_PTR_CONTROL_MEDIA.

Output Param

```
LPWFSPTRREADFORMOUT lpReadFormOut;

typedef struct _wfs_ptr_read_form_out
   {
   LPSTR     lpszFields;
   LPWSTR    lpszUNICODEFields;
   } WFSPTRREADFORMOUT, * LPWFSPTRREADFORMOUT;
```

lpszFields
Pointer to a series of "<FieldName>=<FieldValue>" strings, where each string is null-terminated with the entire field string terminating with two null characters. If the field is an index field, then the syntax of the string is instead "<FieldName>**[**<index>**]**=<FieldValue>", where <index> specifies the zero-based element of the index field.

lpszUNICODEFields
Pointer to a series of "<FieldName>=<FieldValue>" UNICODE strings, where each string is null-terminated with the entire field string terminating with two null characters. If the field is an index field, then the syntax of the string is instead "<FieldName>**[**<index>**]**=<FieldValue>", where <index> specifies the zero-based element of the index field.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_FORMNOTFOUND	The specified form cannot be found.
WFS_ERR_PTR_READNOTSUPPORTED	The device has no read capability.
WFS_ERR_PTR_FIELDSPECFAILURE	The syntax of the *lpszFieldNames* member is invalid.
WFS_ERR_PTR_FIELDERROR	An error occurred while processing a field, causing termination of the print request. An execute event WFS_EXEE_PTR_FIELDERROR is posted with the details.
WFS_ERR_PTR_MEDIANOTFOUND	The specified media definition cannot be found.
WFS_ERR_PTR_MEDIAINVALID	The specified media definition is invalid.
WFS_ERR_PTR_FORMINVALID	The specified form definition is invalid.

WFS_ERR_PTR_MEDIASKEWED	The media skew exceeded the limit in the form definition.
WFS_ERR_PTR_RETRACTBINFULL	The retract bin is full. No more media can be retracted. The current media is still in the device.
WFS_ERR_PTR_SHUTTERFAIL	Open or close of the shutter failed due to manipulation or hardware error.
WFS_ERR_PTR_MEDIAJAMMED	The media is jammed.
WFS_ERR_PTR_INKOUT	No stamping possible, stamping ink supply empty.
WFS_ERR_PTR_LAMPINOP	Imaging lamp is inoperative.
WFS_ERR_PTR_SEQUENCEINVALID	Programming error. Invalid command sequence (e.g. *dwMediaControl* = WFS_PTR_CTRLPARK and park position is busy).
WFS_ERR_PTR_MEDIASIZE	The media entered has an incorrect size.

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_EXEE_PTR_NOMEDIA	No media is present in the device.
WFS_EXEE_PTR_MEDIAINSERTED	Media has been inserted into the device.
WFS_EXEE_PTR_FIELDERROR	A fatal error occurred while processing a field.
WFS_EXEE_PTR_FIELDWARNING	A non-fatal error occurred while processing a field.
WFS_USRE_PTR_RETRACTBINTHRESHOLD	The retract bin is full; operator intervention is required. Note that this event is sent only once, at the point at which the bin becomes full. It is sent with WFS_PTR_RETRACTBINFULL or WFS_PTR_RETRACTBINHIGH status.
WFS_SRVE_PTR_MEDIATAKEN	The media has been taken by the user.
WFS_USRE_PTR_INKTHRESHOLD	The stamping ink supply is low or empty, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_INKLOW or WFS_PTR_INKOUT status.
WFS_USRE_PTR_LAMPTHRESHOLD	The imaging lamp is fading or inoperative, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_LAMPFADING or WFS_PTR_LAMPINOP status.

Comments All error codes (except WFS_ERR_PTR_NOMEDIAPRESENT) and events listed under the WFS_CMD_PTR_CONTROL_MEDIA command description can also occur on this command**.**

The application will use lpszFieldNames or lpszUNICODEFieldNames as an input parameter, depending upon the service provider capabilities. Legacy (non-UNICODE aware) applications will only use the lpszFieldNames field. UNICODE applications can use either the lpszFieldNames or lpszUNICODEFieldNames fields, provided the service provider is UNICODE compliant.
For passbook usage of the lpszFields and lpszUNICODEFields fields the following applies:

If the media type is PASSBOOK, and the field(s) type is TEXT, and the service provider and the underlying passbook printer are capable of detecting available passbook print lines, then the field(s) will be returned without a value, in the format "<FieldName>" or "<FieldName>**[**<index>**]**", if the field is available for passbook printing. Field(s) unavailable for passbook printing will not be returned. The service provider will examine the passbook text field(s) supplied in the lpszFieldNames string, and with the form/fields definition and the underlying passbook printer capability determine which fields should be available for passbook printing.

To illustrate when media type is PASSBOOK, if a form named PSBKTST1 contains 24 fields, one field per line, and the field names are LINE1 through LINE24 (same order as printing), and after execution of this command *lpszFields* contains fields LINE13 through LINE24, then the first print line available for passbook printing is line 13.

To illustrate another example when media type is PASSBOOK, if a form named PSBKTST2 contains 24 fields, one field per line, and the field names are LINE1 through LINE24 (same order as printing), and after execution of this command *lpszFields* contains fields LINE13, and LINE20 through LINE24 then the first print line available for passbook printing is line 13, however lines 14-19 are not also available, so if the application is attempting to determine the first available print line after which all subsequent print lines are also available then line 20 is a better choice.

6.4. WFS_CMD_PTR_RAW_DATA

Description This command is used to send raw data (a byte string of device dependent data) to the physical device.

Input Param

```
LPWFSPTRRAWDATA     lpRawData;
```

```
typedef struct _wfs_ptr_raw_data
   {
   WORD      wInputData;
   ULONG     ulSize;
   LPBYTE    lpbData;
   } WFSPTRRAWDATA, * LPWFSPTRRAWDATA;
```

wInputData
Specifies that input data from the device is expected in response to sending the raw data (i.e., the data contains a command requesting data). Possible values are:

Value	Meaning
WFS_PTR_NOINPUTDATA	No input data is expected.
WFS_PTR_INPUTDATA	Input data is expected.

ulSize
Specifies the size of the byte string passed to the device.

lpbData
Points to the byte string holding the device dependent data.

Output Param

```
LPWFSPTRRAWDATAIN   lpRawDataIn;
```

[used only if *wInputData* is set to WFS_PTR_INPUTDATA]

```
typedef struct _wfs_ptr_raw_data_in
   {
   ULONG     ulSize;
   LPBYTE    lpbData;
   } WFSPTRRAWDATAIN, * LPWFSPTRRAWDATAIN;
```

ulSize
Specifies the size of the byte string received from the device.

lpbData
Points to the byte string received from the device.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_SHUTTERFAIL	Open or close of the shutter failed due to manipulation or hardware error.
WFS_ERR_PTR_MEDIAJAMMED	The media is jammed.
WFS_ERR_PTR_PAPERJAMMED	The paper is jammed.
WFS_ERR_PTR_PAPEROUT	The paper supply is empty.
WFS_ERR_PTR_TONEROUT	Toner or ink supply is empty or printing contrast with ribbon is not sufficient.

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_USRE_PTR_RETRACTBINTHRESHOLD	The retract bin is full; operator intervention is required. Note that this event is sent only once, at the point at which the bin becomes full. It is sent with WFS_PTR_RETRACTBINFULL or WFS_PTR_RETRACTBINHIGH status.
WFS_SRVE_PTR_MEDIATAKEN	The media has been taken by the user.
WFS_USRE_PTR_PAPERTHRESHOLD	The paper supply is low or empty, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_PAPERLOW or WFS_PTR_PAPEROUT status.
WFS_USRE_PTR_TONERTHRESHOLD	The toner or ink supply is low or empty or the printing contrast with ribbon is weak or not sufficient, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_TONERLOW or WFS_PTR_TONEROUT status.

Comments Applications which send raw data to a device will typically not be device or vendor independent. Problems with the use of this command include:

1. The data sent to the device can include commands that change the state of the device in unpredictable ways (in particular, in ways that the service provider may not be aware of).

2. Usage of this command will not be portable.

3. This command violates the XFS forms model that is the basis of XFS printer access.

Thus usage of this command should be avoided whenever possible. If it is used, the usage should be carefully isolated from all other XFS access to the service by at least the **WFSLock** and **WFSUnlock** commands.

6.5. WFS_CMD_PTR_MEDIA_EXTENTS

Description This command is used to get the extents of the media inserted in the physical device. The input parameter specifies the base unit and fractions in which the media extent values will be returned. If no media is present, the device waits for the period of time specified by the *dwTimeOut* parameter in the **WFSExecute** call for media to be inserted.

Input Param

```
LPWFSPTRMEDIAUNIT   lpMediaUnit;

typedef struct _wfs_ptr_media_unit
    {
    WORD          wBase;
    WORD          wUnitX;
    WORD          wUnitY;
    } WFSPTRMEDIAUNIT, * LPWFSPTRMEDIAUNIT;
```

wBase
Specifies the base unit of measurement of the media and can be one of the following values:

Value	Meaning
WFS_FRM_INCH	The base unit is inches.
WFS_FRM_MM	The base unit is millimeters.
WFS_FRM_ROWCOLUMN	The base unit is rows and columns.

wUnitX
Specifies the horizontal resolution of the base units as a fraction of the *wBase* value. For

example, a value of 16 applied to the base unit WFS_FRM_INCH means that the base horizontal resolution is 1/16".
wUnitY
Specifies the vertical resolution of the base units as a fraction of the *wBase* value. For example, a value of 10 applied to the base unit WFS_FRM_MM means that the base vertical resolution is 0.1 mm.

Output Param

```
LPWFSPTRMEDIAEXT    lpMediaExt;
```

```
typedef struct _wfs_ptr_media_ext
    {
    ULONG      ulSizeX;
    ULONG      ulSizeY;
    } WFSPTRMEDIAEXT, * LPWFSPTRMEDIAEXT;
```

ulSizeX
Specifies the width of the media in terms of the base horizontal resolution.
ulSizeY
Specifies the height of the media in terms of the base vertical resolution.

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_EXTENTNOTSUPPORTED	The device cannot report extent(s).
WFS_ERR_PTR_MEDIAJAMMED	The media is jammed.
WFS_ERR_PTR_LAMPINOP	Imaging lamp is inoperative.
WFS_ERR_PTR_MEDIASIZE	The media entered has an incorrect size.

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_EXEE_PTR_NOMEDIA	No media is present in the device.
WFS_EXEE_PTR_MEDIAINSERTED	Media has been inserted into the device.

Comments None.

6.6. WFS_CMD_PTR_RESET_COUNT

Description This function resets the present value for number of media items retracted to zero. The function is possible only for printers with retract capability.

The number of media items retracted is controlled by the service and can be requested before resetting via the info command WFS_INF_PTR_STATUS.

Input Param

```
LPUSHORT      lpusBinNumber;
```

lpusBinNumber
Pointer to the number of the retract bin for which the retract count should be reset to zero. This number has to be between one and the number of bins on the device. If this pointer is NULL all bins will be set to zero.

Output Param None.

Error Codes Only the generic error codes defined in [Ref. 1] can be generated by this command.

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_USRE_PTR_RETRACTBINTHRESHOLD	The status of the retract bin has changed from high or full to a good state. The event is sent with WFS_PTR_RETRACTBINOK status.

Comments None.

6.7. WFS_CMD_PTR_READ_IMAGE

Description This function returns image data from the current media. If no media is present, the device waits for the period of time specified by the *dwTimeOut* parameter in the **WFSExecute** call for media to be inserted.

Input Param

```
LPWFSPTRIMAGEREQUEST       lpImageRequest;
```

```
typedef struct _wfs_ptr_image_request
   {
   WORD         wFrontImageType;
   WORD         wBackImageType;
   WORD         wFrontImageColorFormat;
   WORD         wBackImageColorFormat;
   WORD         wCodelineFormat;
   WORD         fwImageSource;
   LPSTR        lpszFrontImageFile;
   LPSTR        lpszBackImageFile;
   } WFSPTRIMAGEREQUEST, * LPWFSPTRIMAGEREQUEST;
```

wFrontImageType
Specifies the format of the front image returned by this command as one of the following flags (Zero if source not selected):

Value	Meaning
WFS_PTR_IMAGETIF	The returned image is in TIF 6.0 format.
WFS_PTR_IMAGEWMF	The returned image is in WMF (Windows Metafile) format.
WFS_PTR_IMAGEBMP	The returned image is in BMP format.

wBackImageType
Specifies the format of the back image returned by this command as one of the following flags (Zero if source not selected):

Value	Meaning
WFS_PTR_IMAGETIF	The returned image is in TIF 6.0 format.
WFS_PTR_IMAGEWMF	The returned image is in WMF (Windows Metafile) format.
WFS_PTR_IMAGEBMP	The returned image is in BMP format.

wFrontImageColorFormat
Specifies the color format of the requested front image as one of following flags (Zero if source not selected):

Value	Meaning
WFS_PTR_IMAGECOLORBINARY	The scanned images has to be returned in binary (image contains two colors, usually the colors back and white).
WFS_PTR_IMAGECOLORGRAYSCALE	The scanned images has to be returned in gray scale (image contains multiple gray colors).
WFS_PTR_IMAGECOLORFULL	The scanned images has to be returned in full color (image contains colors like red, green, blue etc.).

wBackImageColorFormat
Specifies the color format of the requested back image as one of following flags (Zero if source not selected):

Value	Meaning
WFS_PTR_IMAGECOLORBINARY	The scanned images has to be returned in binary (image contains two colors, usually the colors back and white).
WFS_PTR_IMAGECOLORGRAYSCALE	The scanned images has to be returned in gray scale (image contains multiple gray colors).
WFS_PTR_IMAGECOLORFULL	The scanned images has to be returned in full color (image contains colors like red, green, blue etc.).

wCodelineFormat
Specifies the code line (MICR data) format, as a one of following flags (Zero if source not selected):

Value	Meaning
WFS_PTR_CODELINECMC7	Read CMC7 code line.
WFS_PTR_CODELINEE13B	Read E13B code line.
WFS_PTR_CODELINEOCR	Read code line using OCR.

fwImageSource
Specifies the source as a combination of the following flags:

Value	Meaning
WFS_PTR_IMAGEFRONT	The front image of the document is requested.
WFS_PTR_IMAGEBACK	The back image of the document is requested.
WFS_PTR_CODELINE	The code line of the document is requested.

lpszFrontImageFile
File specifying where to store the front image, e.g. “C:\Temp\FrontImage.bmp”. If a NULL pointer is supplied then the front image data will be returned in the output parameter.
To reduce the size of data sent between the Application and the Service Provider it is recommended to make use of this parameter.

lpszBackImageFile
File specifying where to store the back image, e.g. “C:\Temp\BackImage.bmp”. If a NULL pointer is supplied then the back image data will be returned in the output structure.
To reduce the size of data sent between the Application and the Service Provider it is recommended to make use of this parameter.

Output Param `LPWFSPTRIMAGE *lppImage;`

Pointer to a null-terminated array of pointers to data structures. One array element for each image source requested.

```
typedef struct _wfs_ptr_image
   {
   ~~WORD       wImageType;~~
   ~~ULONG      ulSize;~~
   ~~LPBYTE     lpImage;~~
   WORD       wImageSource;
   WORD       wStatus;
   ULONG      ulDataLength;
   LPBYTE     lpbData;
   } WFSPTRIMAGE, * LPWFSPTRIMAGE;
```

wImageSource
Specifies the source of the data returned by this command as one of the following flags:

Value	Meaning
WFS_PTR_IMAGEFRONT	The front image of the document is requested.
WFS_PTR_IMAGEBACK	The back image of the document is requested.
WFS_PTR_CODELINE	The code line of the document is requested.

wStatus
Status of reading the image data. Possible values are:

Value	Meaning
WFS_PTR_DATAOK	The data is ok.
WFS_PTR_DATASRCNOTSUPP	The data source to read from is not supported by the service provider.
WFS_PTR_DATASRCMISSING	The data source to read from is missing, e.g. the service provider is unable to get the code line.

ulDataLengh
Count of bytes of the following *lpbData*. Zero if the image source is WFS_PTR_IMAGEFRONT or WFS_PTR_IMAGEBACK and the image data has been stored to the hard disk (file name provided).

lpbData
Points to the image or codeline data. NULL pointer if the if the image source is WFS_PTR_IMAGEFRONT or WFS_PTR_IMAGEBACK and the image data has been stored to the hard disk (file name provided).

Error Codes In addition to the generic error codes defined in [Ref. 1], the following error codes can be generated by this command:

Value	Meaning
WFS_ERR_PTR_SHUTTERFAIL	Open or close of the shutter failed due to manipulation or hardware error.
WFS_ERR_PTR_MEDIAJAMMED	The media is jammed. Operator intervention is required.
WFS_ERR_PTR_FILE_IO_ERROR	Directory does not exist or File io error while storing the image to the hard disk.
WFS_ERR_PTR_LAMPINOP	Imaging lamp is inoperative.
WFS_ERR_PTR_MEDIASIZE	The media entered has an incorrect size.

Events In addition to the generic events defined in [Ref. 1], the following events can be generated by this command:

Value	Meaning
WFS_EXEE_PTR_NOMEDIA	No media is present in the device.
WFS_EXEE_PTR_MEDIAINSERTED	Media has been inserted into the device.
WFS_SRVE_PTR_MEDIATAKEN	The media has been taken by the user.
WFS_USRE_PTR_LAMPTHRESHOLD	The imaging lamp is fading or inoperative, operator intervention is required. Note that this event is sent only once, at the point at which the supply becomes low or empty. It is sent with WFS_PTR_LAMPFADING or WFS_PTR_LAMPINOP status.

Comments If the returned image data is in windows bitmap format (BMP) and a file path for storing the image is not supplied, then the first byte of data will be the start of the Bitmap Info Header (this bitmap format is known as DIB, Device Independent Bitmap). The Bitmap File Info Header, which is only present in file versions of bitmaps, will NOT be returned. If the returned image data is in bitmap format (BMP) and a file path for storing the image is supplied, then the first byte of data in the stored file will be the Bitmap File Info Header.

7. New Events

7.1. *WFS_USRE_PTR_LAMPTHRESHOLD*

Description This user event is used to specify that the state of the imaging lamp reached a threshold.

Event Param `LPWORD lpwLampThreshold;`

Specified as one of the following values:

Value	Meaning
WFS_PTR_LAMPOK	The imaging lamp is in a good state.
WFS_PTR_LAMPFADING	The imaging lamp is fading and should be changed.
WFS_PTR_LAMPINOP	The imaging lamp is inoperative.

Comments None.

7.2. *WFS_USRE_PTR_INKTHRESHOLD*

Description This user event is used to specify that the state of the stamping ink reached a threshold.

Event Param `LPWORD lpwInkThreshold;`

Specified as one of the following values:

Value	Meaning
WFS_PTR_INKFULL	The stamping ink in the printer is in a good state.
WFS_PTR_INKLOW	The stamping ink in the printer is low.
WFS_PTR_INKOUT	The stamping ink in the printer is out.

Comments None.

7.3. WFS_SRVE_PTR_MEDIADETECTED

Description This event is generated when a media is detected in the device during a reset operation.

Event Param `LPWFSMEDIADETECTED  lpMediaDetected;`

```
typedef struct _wfs_ptr_media_detected
{
   WORD         wPosition;
   USHORT    usRetractBinNumber;
} WFSPTRMEDIADETECTED, * LPWFSPTRMEDIADETECTED;
```

wPosition

Specifies the media position after the reset operation, as one of the following values:

Value	Meaning
WFS_PTR_MEDIARETRACTED	The media was retracted during the reset operation.
WFS_PTR_MEDIAPRESENT	The media is in the print position or on the stacker.
WFS_PTR_MEDIAENTERING	The media is in the exit slot.
WFS_PTR_MEDIAJAMMED	The media is jammed in the device.
WFS_PTR_MEDIAUNKNOWN	The media is in an unknown position.

usRetractBinNumber
Number of the retract bin the media was retracted to. This number has to be between one and the number of bins supported by this device. It is only relevant if wPosition equals WFS_PTR_MEDIARETRACTED.

Comments None.

8. Changes to existing Events

8.1. WFS_EXEE_PTR_FIELDERROR

Description This event specifies that a fatal error has occurred while processing a field.

Event Param `LPWFSPTRFIELDFAIL   lpFieldFail;`

```
typedef struct _wfs_ptr_field_failure
   {
   LPSTR        lpszFormName;
   LPSTR        lpszFieldName;
   WORD         wFailure;
   } WFSPTRFIELDFAIL, * LPWFSPTRFIELDFAIL;
```

lpszFormName
Points to the null-terminated form name.
lpszFieldName
Points to the null-terminated field name.

wFailure
Specifies the type of failure and can be one of the following values:

Value	Meaning
WFS_PTR_FIELDREQUIRED	The specified field ***must*** be supplied by the application.
WFS_PTR_FIELDSTATICOVWR	The specified field is static and thus ***cannot*** be overwritten by the application.
WFS_PTR_FIELDOVERFLOW	The value supplied for the specified fields is too long.
WFS_PTR_FIELDNOTFOUND	The specified field does not exist.
WFS_PTR_FIELDNOTREAD	The specified field is not an input field.
WFS_PTR_FIELDNOTWRITE	An attempt was made to write to an input field.
WFS_PTR_FIELDHWERROR	The specified field uses special hardware (e.g., OCR) and an error occurred.
WFS_PTR_FIELDTYPENOTSUPPORTED	The form field type is not supported with device.
WFS_PTR_FIELDGRAPHIC	The specified graphic image could not be printed.
WFS_PTR_CHARSETFORM	Service provider does not support character set specified in form.

Comments None.

8.2. *WFS_USRE_PTR_RETRACTBINTHRESHOLD*

Description This event specifies that the status of the retract bin holding the retracted media has changed.

Event Param

```
LPWFSPTRBINTHRESHOLD      lpBinThreshold;
```

```
typedef struct _wfs_ptr_bin_threshold
{
   USHORT    usBinNumber;
   WORD      wRetractBin;
} WFSPTRBINTHRESHOLD, * LPWFSPTRBINTHRESHOLD;
```

usBinNumber
Number of the retract bin for which the status has changed

wRetractBin
Specified as one of the following values:

Value	Meaning
WFS_PTR_RETRACTBINOK	The retract bin of the printer is in a good state.
WFS_PTR_RETRACTBINFULL	The retract bin of the printer is full.
WFS_PTR_RETRACTBINHIGH	The retract bin of the printer is high.

Comments None.

8.3. *WFS_USRE_PTR_PAPERTHRESHOLD*

Description This user event is used to specify that the state of the paper reached a threshold. There is no threshold defined for the parking station as this can contain only one paper item.

Event Param

```
LPWFSPTRPAPERTHRESHOLD     lpPaperThreshold;
```

```
typedef struct _wfs_ptr_paper_threshold
{
   WORD         wPaperSource;
   WORD         wPaperThreshold;
} WFSPTRPAPERTHRESHOLD, * LPWFSPTRPAPERTHRESHOLD;
```

wPaperSource

Specifies the Paper sources as one of the following values:

Value	Meaning
WFS_PTR_PAPERUPPER	An upper paper source is available, devices with only one paper supply must indicate WFS_PTR_PAPERUPPER as being available.
WFS_PTR_PAPERLOWER	A lower paper source is available.
WFS_PTR_PAPEREXTERNAL	An external paper source (such as envelope tray or single sheet feed) is available.
WFS_PTR_PAPERAUX	An auxiliary paper source is available.
WFS_PTR_PAPERAUX2	A second auxiliary paper source is available.

wPaperThreshold

Specified as one of the following values:

Value	Meaning
WFS_PTR_PAPERFULL	The paper in the printer is in a good state.
WFS_PTR_PAPERLOW	The paper in the printer is low.
WFS_PTR_PAPEROUT	The paper in the printer is out.

Comments None.

Implementation Advice:
With version 3.0 there is no need to poll the status of the paper by issuing WFS_INF_PTR_STATUS commands to find out if the paper supply has been refilled. The application may now receive this event if the Service Provider is able to detect the change.

9. Changes to Form, Sub-Form, Field and Media Definitions

This section outlines the format of the definitions of forms, the fields within them, optional tables and fields within the form, and the media on which they are printed.

9.1. Definition Syntax

The syntactic rules for form, field and media definitions are as follows:

• White space	space, tab
• Line continuation	backslash (\)
• Line termination	CR, LF, CR/LF; line termination ends a “keyword section” (a keyword and its value[s])
• Keywords	must be all upper case
• Names	(field/media/font names) any case; case is preserved; service providers are case sensitive
• Strings	all strings must be enclosed in double quote characters ("); standard C escape sequences are allowed.
• Comments	start with two forward slashes (//), end at line termination

Other notes:

- The values of a keyword are separated by commas.
- If a keyword is present, all its values must be specified; default values are used only if the keyword is absent.
- Values that are character strings are marked with asterisks in the definitions below, and must be quoted as specified above.
- The order of attributes within the forms is not mandatory and the attributes may be defined in any order.

- All forms can be represented using either ISO 646 (ANSI) or UNICODE character encoding. If the UNICODE representation is used then all Names and Strings are restricted to an internal representation of ISO 646 (ANSI) characters. Only the INITIALVALUE and FORMAT keyword values can have double byte values outside of the ISO 646 (ANSI) character set.
- If forms character encoding is UNICODE then, consistent with the UNICODE standard, the file prefix must be in little endian (xFFFE) or big endian (xFEFF) notation, such that UNICODE encoding is recognized.

9.2. Form and Media Measurements

The UNIT keyword sections of the form and media definitions specify the base horizontal and vertical resolution as follows:

- the *base* value specifies the base unit of measurement
- the x and y values specify the horizontal and vertical resolution as fractions of the base value (e.g., an *x* value of 10 and a base value of MM means that the base horizontal resolution is 0.1mm).

The base resolutions thus defined by the UNIT keyword section of the ***form*** definition are used as the units of the form definition keyword sections:

- SIZE (*width* and *height* values)
- ALIGNMENT (*xoffset* and *yoffset* values)

and of the sub-form definition keyword sections:

- POSITION (*x* and *y* values)
- SIZE (*width* and *height* values)

and of the field definition keyword sections:

- POSITION (*x* and *y* values)
- SIZE (*width* and *height* values)
- INDEX (*xoffset* and *yoffset* values)

and of the frame definition keyword sections:

- POSITION (*x* and *y* values)
- SIZE (*width* and *height* values)
- REPEATONX (*xoffset* value)
- REPEATONY (*yoffset* value)

The base resolutions thus defined by the UNIT keyword section of the ***media*** definition are used as the units of the media definition keyword sections:

- SIZE (*width* and *height* values)
- PRINTAREA (*x*, *y*, *width* and *height* values)
- RESTRICTED (*x*, *y*, *width* and *height* values)

9.3. Form Definition

XFSFORM		formname*	
BEGIN			
(required)	**UNIT**	base,	Base resolution unit for form definition MM INCH ROWCOLUMN

		x, y	Horizontal base unit fraction Vertical base unit fraction
(required)	**SIZE**	width, height	Width of form Height of form
	ALIGNMENT	alignment, xoffset, yoffset	Alignment of the form on the physical medium: TOPLEFT (default) TOPRIGHT BOTTOMLEFT BOTTOMRIGHT This option allows the positioning of a form onto a physical page relative to any combination of the edges of the physical medium, to support the variations in how devices sense the edge of page for positioning purposes. Horizontal offset relative to the horizontal alignment specified by alignment. Always specified as a positive value (i.e., if aligned to the right side of the medium, means offset the form to the left). (default = 0) Vertical offset relative to the vertical alignment specified by alignment. Always specified as a positive value (i.e., if aligned to the bottom of the medium, means offset the form upward). (default = 0)
	ORIENTATION	type	Orientation of form: PORTRAIT (default) LANDSCAPE
	SKEW	skewfactor	Maximum skew factor in degrees (default = 0)
	VERSION	major, minor, date*, author*	Major version number Minor version number Creation/modification date Author of form
(required)	**LANGUAGE**	languageID	Language used in this form – a 16 bit value (LANGID) which is a combination of a primary (10 bits) and a secondary (6 bits) language ID (This is the standard language ID in the Win32 API; standard macros support construction and decomposition of this composite ID)
	COPYRIGHT	copyright*	Copyright entry
	TITLE	title*	Title of form
	COMMENT	comment*	Comment section
	USERPROMPT	prompt*	Prompt string for user interaction

	[XFSFIELD
	
	BEGIN
	. . .
	END]
		fieldname*	One field definition (as defined in the next section) for each field in the form

	[XFSFRAME
	
	BEGIN
	. . .
	END]
		framename*	One frame definition (as defined in the next section) for each frame in the form

	[XFSSUBFORM
	
	BEGIN
	. . .
	END]
		subformname*	One subform definition (as defined in the next section) for each subform in the form

END			

9.4. SubForm Definition

XFSSUBFORM		subformname*	
BEGIN			
(required)	**POSITION**	X, Y or (Y,Z)	Horizontal position (relative to left side of form) Vertical position (relative to top of form). Format (Y,Z) is used to indicate vertical positioning relative to top of form when top of form is other than 1st page of form, where Z indicates page number (relative to 0) and Y indicates base resolution units relative to top of the form page number (as indicated by Z). Format Y is used to indicate vertical positioning relative to top of the 1st form page.
(required)	**SIZE**	width, height	Width of subform. Width must not exceed width of form. Height of subform. Height must not exceed height of form.

	[XFSFIELD
	BEGIN
	. . .
	END]
		fieldname*	One field definition (as defined in the next section) for each field in the subform

	[XFSFRAME
	BEGIN
	. . .
	END]
		framename*	One frame definition (as defined in the next section) for each frame in the subform

END			

The XFSSUBFORM definition provides a means to isolate a selected area of a form where the user may want to have a select group of fields, frames, and/or running headers and footers. All field and frame definitions within a subform are relative to the POSITION of the subform. A form definition with an imbedded subform will have a series of statements illustrated as follows:

```
XFSFORM
BEGIN
  *
  *
  XFSSUBFORM
  BEGIN
    XFSFIELD
    BEGIN
      *
      *
    END
    XFSFIELD
    BEGIN
      *
      *
    END
  END
END
```

9.5. Field Definition

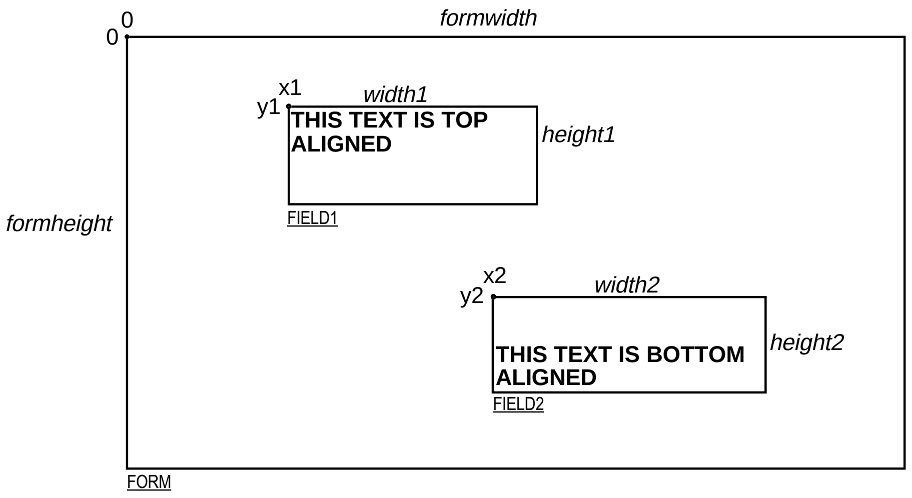
XFSFIELD		fieldname*	
BEGIN			
(required)	**POSITION**	X, Y or (Y,Z)	Horizontal position (relative to left side of form/subform. Vertical position (relative to top of form/subform. Format (Y,Z) is used to indicate vertical positioning relative to top of form/subform when top of form/subform is other than 1st page of form/subform, where Z indicates page number (relative to 0) and Y indicates base resolution units relative to top of the form/subform page number (as indicated by Z). Format Y is used to indicate vertical positioning relative to top of the 1st form/subform.
	FOLLOWS	fieldname*	Print this field directly following the field with the name <fieldname>; positioning information is ignored. See the description of WFS_CMD_PTR_PRINT_FORM. If FOLLOWS is omitted then fields are printed in the order that they appear in the form definition.
	HEADER	 N N-N ALL	This field is either a form/subform header field. N represents a form/subform page number (relative to 0) the header field is to print within. N-N represents a form/subform page number range the header field is to print within. Combinations of N and N-N may exist separated by commas. ALL indicates that header field is to be printed on all pages of form/subform. The form/subform page number is intended to supplement the Z parameter of the POSITION keyword. For example 0,2-4,6 indicates that the header field is to print on relative form/subform pages 0, 2, 3, 4, and 6.
	FOOTER	 N N-N All	This field is either a form/subform footer field. N represents a form/subform page number (relative to 0) the footer field is to print within. N-N represents a form/subform page number range the footer field is to print within. Combinations of N and N-N may exist separated by commas. ALL indicates that footer field is to be printed on all pages of form/subform. The form/subform page number is intended to supplement the Z parameter of the POSITION keyword. For example 0,2-4,6 indicates that the footer field is to print on relative form/subform pages 0, 2, 3, 4, and 6.
	SIDE	side	Side of form where field is positioned: FRONT (default) BACK
(required)	**SIZE**	width, height	Field width Field height
	INDEX	repeatcount,	Count how often this field is repeated in the form, INDEX fields are fixed length. (default is no index field)

		xoffset, yoffset	Horizontal offset for next field Vertical offset for next field
	TYPE	fieldtype	Type of field: TEXT (default) MICR OCR MSF BARCODE GRAPHIC PAGEMARK
	SCALING	scalingtype	Information on how to size the GRAPHIC within the field: BESTFIT (default) scale to size indicated ASIS render at native size MAINTAINASPECT scale as close as possible to size indicated while maintaining the aspect ratio and not losing graphic information. SCALING is only relevant for GRAPHIC field types.
	BARCODE	hriposition	Position of the HRI (Human Readable Interpretation) characters: NONE (default) ABOVE BELOW BOTH The type of barcode to print is defined in the FONT field.
	CLASS	class	Field class OPTIONAL (default) STATIC REQUIRED
	ACCESS	access	Access rights of field WRITE (default) READ READWRITE
	OVERFLOW	overflow	Action on field overflow: TERMINATE (default) TRUNCATE BESTFIT (the service provider fits the data into the field as well as it can) OVERWRITE (a contiguous write) WORDWRAP

	STYLE	style	Display attributes as a combination of the following, ORed together using the "|" operator: NORMAL (default) BOLD ITALIC UNDER (single underline) DOUBLEUNDER (double underline) DOUBLE (double width) TRIPLE (triple width) QUADRUPLE (quadruple width) STRIKETHROUGH ROTATE90 (rotate +90 degrees clockwise) ROTATE270 (rotate +270 degrees clockwise) UPSIDEDOWN (upside down) PROPORTIONAL (proportional spacing) DOUBLEHIGH TRIPLEHIGH QUADRUPLEHIGH CONDENSED SUPERSCRIPT SUBSCRIPT OVERSCORE LETTERQUALITY NEARLETTERQUALITY DOUBLESTRIKE OPAQUE (If omitted then default attribute is transparent) Some of these Styles may be mutually exclusive, or may combine to provide unexpected results.
	CASE	case	Convert field contents to NOCHANGE (default) UPPER LOWER
	HORIZONTAL	justify	Horizontal alignment of field contents LEFT (default) RIGHT CENTER JUSTIFY
	VERTICAL	justify	Vertical alignment of field contents BOTTOM (default) CENTER TOP
	COLOR	color	Color name BLACK (default) WHITE GRAY RED BLUE GREEN YELLOW
	RGBCOLOR	R,G,B	Color in RGB 8 bits per color format. R - The red portion of the RGB value 0-255. G - The green portion of the RGB value 0-255. B - The blue portion of the RGB value 0-255. RBGCOLOR overrides the COLOR attribute.

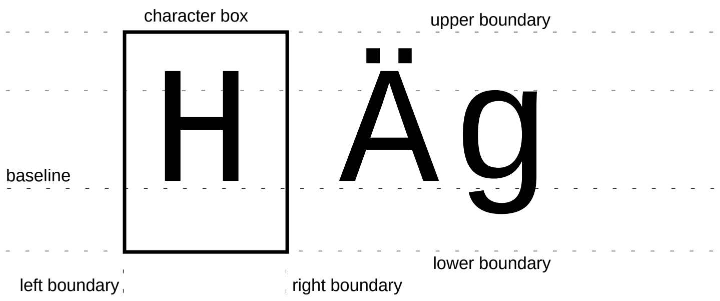
	LANGUAGE	languageID	Language used in this field – a 16 bit value (LANGID) which is a combination of a primary (10 bits) and a secondary (6 bits) language ID (This is the standard language ID in the Win32 API; standard macros support construction and decomposition of this composite ID) If unspecified defaults to form definition LANGUAGE specification.
font	**FONT**	fontname*	Font name: This attribute is interpreted by the service provider. In some cases it may indicate printer resident fonts, and in others it may indicate the name of a downloadable font. For BARCODE fields it represents the barcode font name. In some cases this predefines the following parameters:
definition	**POINTSIZE**	pointsize	Point size
information	**CPI**	cpi	Characters per inch
	LPI	lpi	Lines per inch
	FORMAT	Formatstring*	This is an application defined input field describing how the application should format the data. This may be interpreted by the service provider.
	INITIALVALUE	value*	Initial value, for GRAPHIC type fields, this value may contain the filename of the graphic image. The type of this graphic will be determined by the file extension (e.g. BMP for Windows Bitmap). Graphic file name may be full or partial path. For example “C:\BSVC\BSVCLOGO.BMP” illustrates use of full path name. A file name specification of “LOGO.BMP” illustrates partial path name. In this instance file is obtained from current directory.
END			

The following diagrams illustrate the positioning and sizing of text fields on a form, and, in particular, the vertical alignment of text within a field using **VERTICAL=TOP** and **VERTICAL=BOTTOM** values in the field definition.



VERTICAL=TOP	the upper boundary of the character drawing box (shown below) is positioned vertically to the upper field boundary.
VERTICAL=BOTTOM	the baseline of the character drawing box (shown below) is positioned vertically to the lower field boundary.

Definition of the character drawing box:



When more than one line of text is to be printed in a field, and the definition includes **VERTICAL=BOTTOM,** the vertical position of the first line is calculated using the specified (or implied) **LPI** value.

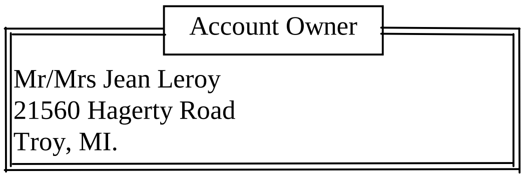
9.6. Frame Definition

XFSFRAME		framename*	
BEGIN			
(required)	**POSITION**	X, Y or (Y,Z)	Horizontal position of left corner of the frame (relative to left side of form/subform. Vertical position of left corner of frame (relative to top of form/subform. Format (Y,Z) is used to indicate vertical positioning of the left corner of the frame relative to top of form/subform when top of form/subform is other than 1st page of form/subform, where Z indicates page number (relative to 0) and Y indicates base resolution units relative to top of the form/subform page number (as indicated by Z). Format Y is used to indicate vertical positioning of the left corner of frame relative to top of the 1st form/subform.
	FRAMES	fieldname*	Frames the field with the name <fieldname>, positioning information is ignored. The frame surrounds the complete field, not just the printed data. If the field is repeated, the frame surrounds the first and last fields that are printed.
	HEADER	 N N-N ALL	This frame is either a form/subform header frame. N represents a form/subform page number (relative to 0) the header frame is to print within. N-N represents a form/subform page number range the header frame is to print within. Combinations of N and N-N may exist separated by commas. ALL indicates that header frame is to be printed on all pages of form/subform. The form/subform page number is intended to supplement the Z parameter of the POSITION keyword. For example 0,2-4,6 indicates that the header frame is to print on relative form/subform pages 0, 2, 3, 4, and 6.
	FOOTER	 N N-N ALL	This field is either a form/subform footer frame. N represents a form/subform page number (relative to 0) the footer frame is to print within. N-N represents a form/subform page number range the footer frame is to print within. Combinations of N and N-N may exist separated by commas. ALL indicates that footer frame is to be printed on all pages of form/subform. The form/subform page number is intended to supplement the Z parameter of the POSITION keyword. For example 0,2-4,6 indicates that the footer frame is to print on relative form/subform pages 0, 2, 3, 4, and 6.
	SIDE	side	Side of form where this frame is positionned: FRONT (default) BACK

(required)	**SIZE**	width, height	Frame width in base horizontal units for the form Frame height in base vertical units for the form
	REPEATONX	repeatcount , xoffset	Count how often this frame is repeated horizontally in the form. Horizontal offset for next frame in base horizontal units.
	REPEATONY	repeatcount , yoffset	Count how often this frame is repeated vertically in the form. Vertical offset for next frame in base vertical units.
	TYPE	frametype	Type of frame: RECTANGLE (default) ROUNDED_CORNER ELLIPSE
	CLASS	class	Frame class: STATIC (default) OPTIONAL (The frame is printed only if its name appears in the list of field names given as parameter to the WFSExecute command. In this case, the name of the frame must be different from all the names of the fields.)
	OVERFLOW	overflow	Action on frame overflowing the form: TERMINATE (default) TRUNCATE BESTFIT (the service provider fits the frame into the media as well as it can)
	STYLE	style	Frame line attributes: SINGLE_THIN (default) DOUBLE_THIN SINGLE_THICK DOUBLE_THICK DOTTED
	COLOR	color	Color name for frame lines: BLACK (default) WHITE GRAY RED BLUE GREEN YELLOW
	RGBCOLOR	R,G,B	Color in RGB 8 bits per color format. R - The red portion of the RGB value 0-255. G - The green portion of the RGB value 0-255. B - The blue portion of the RGB value 0-255. RGBCOLOR overrides the COLOR attribute.
	FILLCOLOR	color	Color name for interior of frame: BLACK WHITE (default) GRAY RED BLUE GREEN YELLOW

	RGBFILLCOLOR	R,G,B	Color in RGB 8 bits per color format. R - The red portion of the RGB value 0-255. G - The green portion of the RGB value 0-255. B - The blue portion of the RGB value 0-255. RGBFILLCOLOR overrides the FILLCOLOR attribute.
	FILLSTYLE	style	Style for filling the interior of frame: NONE (default) SOLID Solid color BDIAGONAL Downward hatch (left to right) at 45 degrees CROSS Horizontal and vertical crosshatch DIAGCROSS Crosshatch at 45 degrees FDIAGONAL Upward hatch (left to right) at 45 degrees HORIZONTAL Horizontal hatch VERTICAL Vertical hatch
	SUBSTSIGN	substitute sign	Character that is used as substitue sign when a character in a read field cannot be read
frame title	**TITLE**	fieldname*	Uses the field with the name <fieldname> as the title of the frame. Positioning information of the field is ignored.
definition	**HORIZONTAL**	justify	Horizontal alignment of the frame title: LEFT (default) CENTER RIGHT
information	**VERTICAL**	justify	Vertical alignment of the frame title: TOP (default) BOTTOM
END			

The XFSFRAME definition provides a means for framing a XFSFIELD text field. The basic concept of a XFSFRAME definition and corresponding XFSFIELD definition is illustrated as follows:



When the **XFSFRAME** frames a field, its positioning and size information are ignored. Instead, service providers should position the top left corner of the frame one horizontal base unit to the left and one vertical base unit to the top of the top left corner of the field. Similarly, service providers should size the frame so that it bottom right corner is one base unit below and to the right to the field. For instance, if the form units are **ROWCOLUMN**, and a **XFSFRAME** “A” is said to **FRAME** the **XFSFIELD** “B” which is positioned at row 1, column 1 with a size of 1 row and 20 columns, the frame will be drawn from row 0, column 0 to row 3, column 22.

The horizontal and vertical positioning of a frame title override the position of the named **XFSFIELD**. For instance, if a **XFSFRAME** “A” is said to have the **XFSFIELD** “B” as its title, with the default horizontal and vertical title justification, it is just as if **XFSFIELD** “B” had been positioned at the top left corner of the frame. Note that the **SIZE** information for the title field still is meaningful: it gives the starting and/or ending positions of the frame lines.

The **SIDE** attributes of the **XFSFRAME** and the **XFSFIELD**s it refers to must agree.

The width of the lines and the interval between the lines of doubled frames are vendor specific. Whether the lines are drawn using graphics printing or using pseudo-graphic is vendor specific. However, service provider are responsible for rendering intersecting frames.

Depending on the printer technology, framing of fields can substantially slow down the print process.

Support of framing by a service provider or the device it controls is not mandatory to be XFS compliant.

Sample 1: Simple framing

```
XFSFORM "Multiple Balances"
BEGIN
  UNIT INCH, 16, 16
  SIZE 91, 64
  VERSION 1, 0, "13/09/96", "XFS"
  LANGUAGE 0x0409
XFSFIELD "Account Title"
  BEGIN
    POSITION 15, 4
    SIZE 30, 4
    CLASS STATIC
    HORIZONTAL CENTER
    INITIALVALUE "Account"
  END
  XFSFIELD "Balance Title"
  BEGIN
    POSITION 45, 4
    SIZE 30, 4
    CLASS STATIC
    HORIZONTAL CENTER
    INITIALVALUE "Balance"
  END

  XFSFIELD "Account"
  BEGIN
    POSITION 15, 8
```

When printed with the following field list:
Account[0]=0123456789123001
Account[1]=0123456789123002
Account[2]=0123456789123003
Balance[0]=$17465.12
Balance[1] =$2458.23
Balance[2]=$6542.78
Will print:

Account	Balance
0123456789123001 0123456789123002 0123456789123003	$17465.12 $2458.23 $6542.78

When printed with the following field list:
Account[0]=0123456789123001
Balance[0]=$17465.12
Will print:

Account	Balance
0123456789123001	$17465.12

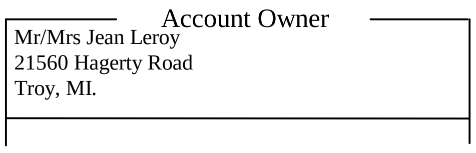
```
      SIZE 30, 4
      INDEX 10, 0, 3
    END //"Account"
    XFSFIELD "Balance"
    BEGIN
      POSITION 45, 8
      SIZE 30, 4
      INDEX 10, 0, 3
      HORIZONTAL RIGHT
    END //"Balance"
    XFSFRAME "Account Title"
    BEGIN
      POSITION 15, 4
      FRAMES "Account Title"
      SIZE 30, 4
      STYLE DOUBLE_THIN
    END
    XFSFRAME "Balance Title"
    BEGIN
       POSITION 45, 4
       FRAMES "Balance Title"
       SIZE 30, 4
       STYLE DOUBLE_THIN
    END
    XFSFRAME "Account"
    BEGIN
       POSITION 15, 8
       FRAMES "Account"
       SIZE 30, 34
       STYLE DOUBLE_THIN
    END
    XFSFRAME "Balance"
    BEGIN
       POSITION 45, 8
       FRAMES "Balance"
       SIZE 30, 34
       STYLE DOUBLE_THIN
    END
  END
```

Sample 2: Framing with title

```
  XFSFORM "Bank Details"
  BEGIN
    UNIT INCH, 16, 16
    SIZE 121, 64
    VERSION 1, 0, "13/09/96", "XFS Editor"
    LANGUAGE 0x0409
    XFSFIELD "Owner Frame Title"
    BEGIN
      POSITION 24, 9
      SIZE 27, 3
      CLASS STATIC
      HORIZONTAL CENTER
      VERTICAL CENTER
      INITIALVALUE "Account Owner"
     END
     XFSFIELD "Owner"
     BEGIN
        POSITION 20, 11
        SIZE 35, 9
        CLASS REQUIRED
        VERTICAL TOP
    END //"Owner"
     XFSFRAME "Owner Frame"
     BEGIN
        POSITION 19, 10
        FRAMES "Owner"
        SIZE 37, 11
        TITLE "Owner Frame Title"
        HORIZONTAL CENTER
    END
  END
```

When printed with the following field list:
Owner = Mr/Mrs Jean Leroy
            21560 Hagerty Road
            Troy, MI.

will print:


Account Owner
Mr/Mrs Jean Leroy
21560 Hagerty Road
Troy, MI.

Sample 3: Framing with filled interior

```
XFSFORM "Bank Details"
BEGIN
   UNIT INCH, 16, 16
   SIZE 121, 64
   VERSION 1, 0, "13/09/96", "XFS Editor"
   LANGUAGE 0x0409
   XFSFIELD "Owner"
   BEGIN
      POSITION 20, 11
      SIZE 35, 9
      CLASS REQUIRED
       VERTICAL TOP
   END
   XFSFRAME "Owner Frame"
   BEGIN
       POSITION 19, 10
       FRAMES "Owner"
       SIZE 37, 11
       FILLCOLOR GRAY
       FILLSTYLE CROSS
   END
END
```

When printed with the following field list:

```
Owner = Mr/Mrs Jean Leroy
             21560 Hagerty Road
             Troy, MI.
```

will print:

```
Mr/Mrs Jean Leroy
21560 Hagerty Road
Troy, MI.
```

Sample 4: Repeated Framing

```
XFSFORM "Smart Account Number"
BEGIN
   UNIT INCH, 16, 16
   SIZE 121, 64
   VERSION 1, 0, "13/09/96", "XFS Editor"
    LANGUAGE 0x0409
   XFSFIELD "Account Number"
   BEGIN
      POSITION 20, 8
      SIZE 4, 4
      INDEX 12, 4, 0
      HORIZONTAL CENTER
      VERTICAL CENTER
   END
   XFSFRAME "A/N Frame"
   BEGIN
       POSITION 20, 8
       SIZE 4, 4
       REPEATONX 12, 4
    END
END
```

When printed with the following field list:

```
Account Number[0]=0
Account Number[1]=1
Account Number[2]=2
Account Number[3]=3
Account Number[4]=4
Account Number[5]=5
Account Number[6]=6
Account Number[7]=7
Account Number[8]=8
Account Number[9]=9
Account Number[10]=0
Account Number[11]=1
```

will print

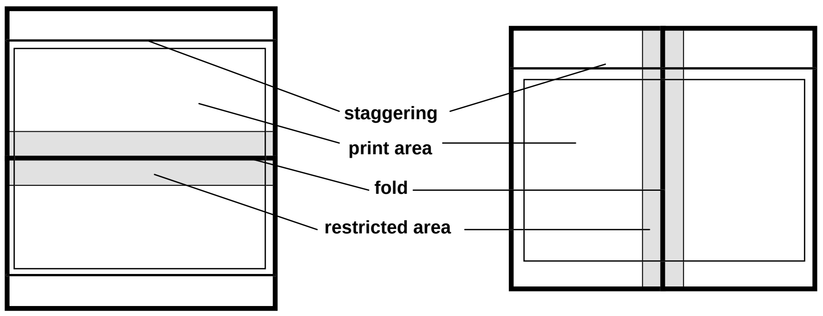
0	1	2	3	4	5	6	7	8	9	0	1

9.7. Media Definition

The media definition determines those characteristics that result from the combination of a particular media type together with a particular vendor's printer. The aim is to make it easy to move forms between different vendors' printers which might have different constraints on how they handle a specific media type. It is the service provider's responsibility to ensure that the form definition does not specify the printing of any fields that conflict with the media definition. An example of such a conflict might be that the form definition asks for a field to be printed in an area that the media definition defines as an unprintable area.

The media definition is also intended to provide the capability of defining media types that are specific to the financial industry. An example is a passbook as shown below.

Passbook with horizontal fold **Passbook with vertical fol**



XFSMEDIA		medianame*	
BEGIN			
	TYPE	type	Predefined media types are: GENERIC (default) MULTIPART PASSBOOK
	SOURCE	source	Paper source: ANY (default) UPPER LOWER EXTERNAL (envelope tray or single sheet feed tray} AUX
(required)	**UNIT**	base, x, y,	Base resolution unit for media definition MM INCH ROWCOLUMN Horizontal base unit fraction Vertical base unit fraction
(required)	**SIZE**	width, height	Width of physical media Height of physical media (0 = unlimited, i.e, roll paper)
	PRINTAREA	x, y, width, height	Printable area relative to top left corner of physical media (default = physical size of media)
	RESTRICTED	x, y, width, height	Restricted area relative to to top left corner of physical media (default = no restricted area)
	FOLD	fold	Type of passbook HORIZONTAL VERTICAL
	STAGGERING	staggering	Staggering of passbook from top (default = 0)
	PAGE	count	Number of pages in passbook (default = 0)
	LINES	count	Number of printable lines (default = 0)
END			

10. Changes to C-Header file

```
/******************************************************************************
*                                                                             *
* xfsptr.h    XFS - Banking Printer (PTR) definitions                         *
*               (receipt, journal, passbook and document printer)            *
*                                                                             *
*             Version 3.00  (10/18/2000)                                      *
*                                                                             *
******************************************************************************/

#ifndef __INC_XFSPTR__H
#define __INC_XFSPTR__H

#ifdef __cplusplus
extern "C" {
#endif

#include <xfsapi.h>

/*   be aware of alignment   */
#pragma pack(push,1)


/* value of WFSPTRCAPS.wClass */

#define     WFS_SERVICE_CLASS_PTR               (1)
#define     WFS_SERVICE_CLASS_VERSION_PTR       (0x0003) /* Version 3.00 */
#define     WFS_SERVICE_CLASS_NAME_PTR          "PTR"

#define     PTR_SERVICE_OFFSET                  (WFS_SERVICE_CLASS_PTR * 100)


/* PTR Info Commands */

#define     WFS_INF_PTR_STATUS                  (PTR_SERVICE_OFFSET + 1)
#define     WFS_INF_PTR_CAPABILITIES            (PTR_SERVICE_OFFSET + 2)
#define     WFS_INF_PTR_FORM_LIST               (PTR_SERVICE_OFFSET + 3)
#define     WFS_INF_PTR_MEDIA_LIST              (PTR_SERVICE_OFFSET + 4)
#define     WFS_INF_PTR_QUERY_FORM              (PTR_SERVICE_OFFSET + 5)
#define     WFS_INF_PTR_QUERY_MEDIA             (PTR_SERVICE_OFFSET + 6)
#define     WFS_INF_PTR_QUERY_FIELD             (PTR_SERVICE_OFFSET + 7)


/* PTR Execute Commands */

#define     WFS_CMD_PTR_CONTROL_MEDIA           (PTR_SERVICE_OFFSET + 1)
#define     WFS_CMD_PTR_PRINT_FORM              (PTR_SERVICE_OFFSET + 2)
#define     WFS_CMD_PTR_READ_FORM               (PTR_SERVICE_OFFSET + 3)
#define     WFS_CMD_PTR_RAW_DATA                (PTR_SERVICE_OFFSET + 4)
#define     WFS_CMD_PTR_MEDIA_EXTENTS           (PTR_SERVICE_OFFSET + 5)
#define     WFS_CMD_PTR_RESET_COUNT             (PTR_SERVICE_OFFSET + 6)
#define     WFS_CMD_PTR_READ_IMAGE              (PTR_SERVICE_OFFSET + 7)
#define     WFS_CMD_PTR_RESET                   (PTR_SERVICE_OFFSET + 8)
#define     WFS_CMD_PTR_RETRACT_MEDIA           (PTR_SERVICE_OFFSET + 9)
#define     WFS_CMD_PTR_DISPENSE_PAPER          (PTR_SERVICE_OFFSET + 10)


/* PTR Messages */

#define     WFS_EXEE_PTR_NOMEDIA                (PTR_SERVICE_OFFSET + 1)
#define     WFS_EXEE_PTR_MEDIAINSERTED          (PTR_SERVICE_OFFSET + 2)
#define     WFS_EXEE_PTR_FIELDERROR             (PTR_SERVICE_OFFSET + 3)
#define     WFS_EXEE_PTR_FIELDWARNING           (PTR_SERVICE_OFFSET + 4)
#define     WFS_USRE_PTR_RETRACTBINTHRESHOLD    (PTR_SERVICE_OFFSET + 5)
#define     WFS_SRVE_PTR_MEDIATAKEN             (PTR_SERVICE_OFFSET + 6)
#define     WFS_USRE_PTR_PAPERTHRESHOLD         (PTR_SERVICE_OFFSET + 7)
#define     WFS_USRE_PTR_TONERTHRESHOLD         (PTR_SERVICE_OFFSET + 8)
#define     WFS_SRVE_PTR_MEDIAINSERTED          (PTR_SERVICE_OFFSET + 9)
#define     WFS_USRE_PTR_LAMPTHRESHOLD          (PTR_SERVICE_OFFSET + 10)
#define     WFS_USRE_PTR_INKTHRESHOLD           (PTR_SERVICE_OFFSET + 11)
```

```
#define     WFS_SRVE_PTR_MEDIADETECTED         (PTR_SERVICE_OFFSET + 12)


/* values of WFSPTRSTATUS.fwDevice */

#define     WFS_PTR_DEVONLINE                  WFS_STAT_DEVONLINE
#define     WFS_PTR_DEVOFFLINE                 WFS_STAT_DEVOFFLINE
#define     WFS_PTR_DEVPOWEROFF                WFS_STAT_DEVPOWEROFF
#define     WFS_PTR_DEVNODEVICE                WFS_STAT_DEVNODEVICE
#define     WFS_PTR_DEVHWERROR                 WFS_STAT_DEVHWERROR
#define     WFS_PTR_DEVUSERERROR               WFS_STAT_DEVUSERERROR
#define     WFS_PTR_DEVBUSY                    WFS_STAT_DEVBUSY


/* values of WFSPTRSTATUS.fwMedia and
             WFSPTRMEDIADETECTED.wPosition */

#define     WFS_PTR_MEDIAPRESENT               (0)
#define     WFS_PTR_MEDIANOTPRESENT            (1)
#define     WFS_PTR_MEDIAJAMMED                (2)
#define     WFS_PTR_MEDIANOTSUPP               (3)
#define     WFS_PTR_MEDIAUNKNOWN               (4)
#define     WFS_PTR_MEDIAENTERING              (5)
#define     WFS_PTR_MEDIARETRACTED             (6)


/* Size and max index of fwPaper array */

#define     WFS_PTR_SUPPLYSIZE     (16)
#define     WFS_PTR_SUPPLYMAX      (WFS_PTR_SUPPLYSIZE - 1)


/* Indices of WFSPTRSTATUS.fwPaper [...] */

#define     WFS_PTR_SUPPLYUPPER                (0)
#define     WFS_PTR_SUPPLYLOWER                (1)
#define     WFS_PTR_SUPPLYEXTERNAL             (2)
#define     WFS_PTR_SUPPLYAUX                  (3)
#define     WFS_PTR_SUPPLYAUX2                 (4)
#define     WFS_PTR_SUPPLYPARK                 (5)


/* values of WFSPTRSTATUS.fwPaper and
             WFSPTRPAPERTHRESHOLD.wPaperThreshold */

#define     WFS_PTR_PAPERFULL                  (0)
#define     WFS_PTR_PAPERLOW                   (1)
#define     WFS_PTR_PAPEROUT                   (2)
#define     WFS_PTR_PAPERNOTSUPP               (3)
#define     WFS_PTR_PAPERUNKNOWN               (4)
#define     WFS_PTR_PAPERJAMMED                (5)


/* values of WFSPTRSTATUS.fwToner */

#define     WFS_PTR_TONERFULL                  (0)
#define     WFS_PTR_TONERLOW                   (1)
#define     WFS_PTR_TONEROUT                   (2)
#define     WFS_PTR_TONERNOTSUPP               (3)
#define     WFS_PTR_TONERUNKNOWN               (4)


/* values of WFSPTRSTATUS.fwInk */

#define     WFS_PTR_INKFULL                    (0)
#define     WFS_PTR_INKLOW                     (1)
#define     WFS_PTR_INKOUT                     (2)
#define     WFS_PTR_INKNOTSUPP                 (3)
#define     WFS_PTR_INKUNKNOWN                 (4)


/* values of WFSPTRSTATUS.fwLamp */

#define     WFS_PTR_LAMPOK                     (0)
```

```
#define    WFS_PTR_LAMPFADING                (1)
#define    WFS_PTR_LAMPINOP                  (2)
#define    WFS_PTR_LAMPNOTSUPP               (3)
#define    WFS_PTR_LAMPUNKNOWN               (4)


/* values of WFSPTRSTATUS.fwRetractBin and
             WFSPTRBINTHRESHOLD.wRetractBin */

#define    WFS_PTR_RETRACTBINOK              (0)
#define    WFS_PTR_RETRACTBINFULL            (1)
#define    WFS_PTR_RETRACTNOTSUPP            (2)
#define    WFS_PTR_RETRACTUNKNOWN            (3)
#define    WFS_PTR_RETRACTBINHIGH            (4)


/* values of WFSPTRCAPS.fwType */

#define    WFS_PTR_TYPERECEIPT               0x0001
#define    WFS_PTR_TYPEPASSBOOK              0x0002
#define    WFS_PTR_TYPEJOURNAL               0x0004
#define    WFS_PTR_TYPEDOCUMENT              0x0008
#define    WFS_PTR_TYPESCANNER               0x0010


/* values of WFSPTRCAPS.wResolution, WFSPTRPRINTFORM.wResolution */

#define    WFS_PTR_RESLOW                    0x0001
#define    WFS_PTR_RESMED                    0x0002
#define    WFS_PTR_RESHIGH                   0x0004
#define    WFS_PTR_RESVERYHIGH               0x0008


/* values of WFSPTRCAPS.fwReadForm */

#define    WFS_PTR_READOCR                   0x0001
#define    WFS_PTR_READMICR                  0x0002
#define    WFS_PTR_READMSF                   0x0004
#define    WFS_PTR_READBARCODE               0x0008
#define    WFS_PTR_READPAGEMARK              0x0010
#define    WFS_PTR_READIMAGE                 0x0020
#define    WFS_PTR_READEMPTYLINE             0x0040


/* values of WFSPTRCAPS.fwWriteForm */

#define    WFS_PTR_WRITETEXT                 0x0001
#define    WFS_PTR_WRITEGRAPHICS             0x0002
#define    WFS_PTR_WRITEOCR                  0x0004
#define    WFS_PTR_WRITEMICR                 0x0008
#define    WFS_PTR_WRITEMSF                  0x0010
#define    WFS_PTR_WRITEBARCODE              0x0020
#define    WFS_PTR_WRITESTAMP                0x0040


/* values of WFSPTRCAPS.fwExtents */

#define    WFS_PTR_EXTHORIZONTAL             0x0001
#define    WFS_PTR_EXTVERTICAL               0x0002


/* values of WFSPTRCAPS.fwControl, dwMediaControl */

#define    WFS_PTR_CTRLEJECT                 0x0001
#define    WFS_PTR_CTRLPERFORATE             0x0002
#define    WFS_PTR_CTRLCUT                   0x0004
#define    WFS_PTR_CTRLSKIP                  0x0008
#define    WFS_PTR_CTRLFLUSH                 0x0010
#define    WFS_PTR_CTRLRETRACT               0x0020
#define    WFS_PTR_CTRLSTACK                 0x0040
#define    WFS_PTR_CTRLPARTIALCUT            0x0080
#define    WFS_PTR_CTRLALARM                 0x0100
#define    WFS_PTR_CTRLATPFORWARD            0x0200
#define    WFS_PTR_CTRLATPBACKWARD           0x0400
```

```
#define     WFS_PTR_CTRLTURNMEDIA               0x0800
#define     WFS_PTR_CTRLSTAMP                   0x1000
#define     WFS_PTR_CTRLPARK                    0x2000


/* values of WFSPTRCAPS.fwPaperSources,
             WFSFRMMEDIA.wPaperSources,
             WFSPTRPRINTFORM.wPaperSource and
             WFSPTRPAPERTHRESHOLD.wPaperSource   */

#define     WFS_PTR_PAPERANY                    0x0001
#define     WFS_PTR_PAPERUPPER                  0x0002
#define     WFS_PTR_PAPERLOWER                  0x0004
#define     WFS_PTR_PAPEREXTERNAL               0x0008
#define     WFS_PTR_PAPERAUX                    0x0010
#define     WFS_PTR_PAPERAUX2                   0x0020
#define     WFS_PTR_PAPERPARK                   0x0040


/* values of WFSPTRCAPS.fwImageType,
             WFSPTRIMAGEREQUEST.wFrontImageFormat and
             WFSPTRIMAGEREQUEST.wBackImageFormat */

#define     WFS_PTR_IMAGETIF                    0x0001
#define     WFS_PTR_IMAGEWMF                    0x0002
#define     WFS_PTR_IMAGEBMP                    0x0004


/* values of WFSPTRCAPS.fwFrontImageColorFormat,
             WFSPTRCAPS.fwBackImageColorFormat,
             WFSPTRIMAGEREQUEST.wFrontImageColorFormat and
             WFSPTRIMAGEREQUEST.wBackImageColorFormat */

#define     WFS_PTR_IMAGECOLORBINARY            0x0001
#define     WFS_PTR_IMAGECOLORGRAYSCALE         0x0002
#define     WFS_PTR_IMAGECOLORFULL              0x0004


/* values of WFSPTRCAPS.fwCodelineFormat and
             WFSPTRIMAGEREQUEST.wCodelineFormat */

#define     WFS_PTR_CODELINECMC7                0x0001
#define     WFS_PTR_CODELINEE13B                0x0002
#define     WFS_PTR_CODELINEOCR                 0x0004


/* values of WFSPTRCAPS.fwImageSource,
             WFSPTRIMAGEREQUEST.fwImageSource and
             WFSPTRIMAGE.wImageSource */

#define     WFS_PTR_IMAGEFRONT                  0x0001
#define     WFS_PTR_IMAGEBACK                   0x0002
#define     WFS_PTR_CODELINE                    0x0004


/* values of WFSPTRCAPS.fwCharSupport, WFSFRMHEADER.fwCharSupport  */

#define     WFS_PTR_ASCII                       0x0001
#define     WFS_PTR_UNICODE                     0x0002


/* values of WFSFRMHEADER.wBase, WFSFRMMEDIA.wBase, WFSPTRMEDIAUNIT.wBase */

#define     WFS_FRM_INCH                        (0)
#define     WFS_FRM_MM                          (1)
#define     WFS_FRM_ROWCOLUMN                   (2)


/* values of WFSFRMHEADER.wAlignment */

#define     WFS_FRM_TOPLEFT                     (0)
#define     WFS_FRM_TOPRIGHT                    (1)
#define     WFS_FRM_BOTTOMLEFT                  (2)
#define     WFS_FRM_BOTTOMRIGHT                 (3)
```

```
/* values of WFSFRMHEADER.wOrientation */

#define     WFS_FRM_PORTRAIT                 (0)
#define     WFS_FRM_LANDSCAPE                (1)


/* values of WFSFRMMEDIA.fwMediaType */

#define     WFS_FRM_MEDIAGENERIC             (0)
#define     WFS_FRM_MEDIAPASSBOOK            (1)
#define     WFS_FRM_MEDIAMULTIPART           (2)


/* values of WFSFRMMEDIA.fwFoldType */

#define     WFS_FRM_FOLDNONE                 (0)
#define     WFS_FRM_FOLDHORIZONTAL           (1)
#define     WFS_FRM_FOLDVERTICAL             (2)


/* values of WFSFRMFIELD.fwType */

#define     WFS_FRM_FIELDTEXT                (0)
#define     WFS_FRM_FIELDMICR                (1)
#define     WFS_FRM_FIELDOCR                 (2)
#define     WFS_FRM_FIELDMSF                 (3)
#define     WFS_FRM_FIELDBARCODE             (4)
#define     WFS_FRM_FIELDGRAPHIC             (5)
#define     WFS_FRM_FIELDPAGEMARK            (6)


/* values of WFSFRMFIELD.fwClass */

#define     WFS_FRM_CLASSSTATIC              (0)
#define     WFS_FRM_CLASSOPTIONAL            (1)
#define     WFS_FRM_CLASSREQUIRED            (2)


/* values of WFSFRMFIELD.fwAccess */

#define     WFS_FRM_ACCESSREAD               0x0001
#define     WFS_FRM_ACCESSWRITE              0x0002


/* values of WFSFRMFIELD.fwOverflow */

#define     WFS_FRM_OVFTERMINATE             (0)
#define     WFS_FRM_OVFTRUNCATE              (1)
#define     WFS_FRM_OVFBESTFIT               (2)
#define     WFS_FRM_OVFOVERWRITE             (3)
#define     WFS_FRM_OVFWORDWRAP              (4)


/* values of WFSPTRFIELDFAIL.wFailure */

#define     WFS_PTR_FIELDREQUIRED            (0)
#define     WFS_PTR_FIELDSTATICOVWR          (1)
#define     WFS_PTR_FIELDOVERFLOW            (2)
#define     WFS_PTR_FIELDNOTFOUND            (3)
#define     WFS_PTR_FIELDNOTREAD             (4)
#define     WFS_PTR_FIELDNOTWRITE            (5)
#define     WFS_PTR_FIELDHWERROR             (6)
#define     WFS_PTR_FIELDTYPENOTSUPPORTED    (7)
#define     WFS_PTR_FIELDGRAPHIC             (8)
#define     WFS_PTR_CHARSETFORM              (9)


/* values of WFSPTRPRINTFORM.wAlignment */

#define     WFS_PTR_ALNUSEFORMDEFN           (0)
#define     WFS_PTR_ALNTOPLEFT               (1)
#define     WFS_PTR_ALNTOPRIGHT              (2)
```

```
#define    WFS_PTR_ALNBOTTOMLEFT            (3)
#define    WFS_PTR_ALNBOTTOMRIGHT           (4)


/* values of WFSPTRPRINTFORM.wOffsetX and WFSPTRPRINTFORM.wOffsetY */

#define    WFS_PTR_OFFSETUSEFORMDEFN        0xffff


/* values of WFSPTRRAWDATA.wInputData */

#define    WFS_PTR_NOINPUTDATA              (0)
#define    WFS_PTR_INPUTDATA                (1)


/* values of WFSPTRIMAGE.wStatus */

#define     WFS_PTR_DATAOK                  (0)
#define     WFS_PTR_DATASRCNOTSUPP          (1)
#define     WFS_PTR_DATASRCMISSING          (2)


/* XFS PTR Errors */

#define    WFS_ERR_PTR_FORMNOTFOUND         (-(PTR_SERVICE_OFFSET + 0))
#define    WFS_ERR_PTR_FIELDNOTFOUND        (-(PTR_SERVICE_OFFSET + 1))
#define    WFS_ERR_PTR_NOMEDIAPRESENT       (-(PTR_SERVICE_OFFSET + 2))
#define    WFS_ERR_PTR_READNOTSUPPORTED     (-(PTR_SERVICE_OFFSET + 3))
#define    WFS_ERR_PTR_FLUSHFAIL            (-(PTR_SERVICE_OFFSET + 4))
#define    WFS_ERR_PTR_MEDIAOVERFLOW        (-(PTR_SERVICE_OFFSET + 5))
#define    WFS_ERR_PTR_FIELDSPECFAILURE     (-(PTR_SERVICE_OFFSET + 6))
#define    WFS_ERR_PTR_FIELDERROR           (-(PTR_SERVICE_OFFSET + 7))
#define    WFS_ERR_PTR_MEDIANOTFOUND        (-(PTR_SERVICE_OFFSET + 8))
#define    WFS_ERR_PTR_EXTENTNOTSUPPORTED   (-(PTR_SERVICE_OFFSET + 9))
#define    WFS_ERR_PTR_MEDIAINVALID         (-(PTR_SERVICE_OFFSET + 10))
#define    WFS_ERR_PTR_FORMINVALID          (-(PTR_SERVICE_OFFSET + 11))
#define    WFS_ERR_PTR_FIELDINVALID         (-(PTR_SERVICE_OFFSET + 12))
#define    WFS_ERR_PTR_MEDIASKEWED          (-(PTR_SERVICE_OFFSET + 13))
#define    WFS_ERR_PTR_RETRACTBINFULL       (-(PTR_SERVICE_OFFSET + 14))
#define    WFS_ERR_PTR_STACKERFULL          (-(PTR_SERVICE_OFFSET + 15))
#define    WFS_ERR_PTR_PAGETURNFAIL         (-(PTR_SERVICE_OFFSET + 16))
#define    WFS_ERR_PTR_MEDIATURNFAIL        (-(PTR_SERVICE_OFFSET + 17))
#define    WFS_ERR_PTR_SHUTTERFAIL          (-(PTR_SERVICE_OFFSET + 18))
#define    WFS_ERR_PTR_MEDIAJAMMED          (-(PTR_SERVICE_OFFSET + 19))
#define    WFS_ERR_PTR_FILE_IO_ERROR        (-(PTR_SERVICE_OFFSET + 20))
#define    WFS_ERR_PTR_CHARSETDATA          (-(PTR_SERVICE_OFFSET + 21))
#define    WFS_ERR_PTR_PAPERJAMMED          (-(PTR_SERVICE_OFFSET + 22))
#define    WFS_ERR_PTR_PAPEROUT             (-(PTR_SERVICE_OFFSET + 23))
#define    WFS_ERR_PTR_INKOUT               (-(PTR_SERVICE_OFFSET + 24))
#define    WFS_ERR_PTR_TONEROUT             (-(PTR_SERVICE_OFFSET + 25))
#define    WFS_ERR_PTR_LAMPINOP             (-(PTR_SERVICE_OFFSET + 26))
#define    WFS_ERR_PTR_SOURCEINVALID        (-(PTR_SERVICE_OFFSET + 27))
#define    WFS_ERR_PTR_SEQUENCEINVALID      (-(PTR_SERVICE_OFFSET + 28))
#define    WFS_ERR_PTR_MEDIASIZE            (-(PTR_SERVICE_OFFSET + 29))


/*=================================================================*/
/* PTR Info Command Structures */
/*=================================================================*/

typedef struct _wfs_ptr_retract_bins
{
   WORD      wRetractBin;
   USHORT    usRetractCount;
} WFSPTRRETRACTBINS, * LPWFSPTRRETRACTBINS;

typedef struct _wfs_ptr_status
{
    WORD            fwDevice;
    WORD            fwMedia;
    WORD            fwPaper[WFS_PTR_SUPPLYSIZE];
    WORD            fwToner;
    WORD            fwInk;
    WORD            fwLamp;
    LPWFSPTRRETRACTBINS *lppRetractBins;
    USHORT          usMediaOnStacker;
    LPSTR           lpszExtra;
} WFSPTRSTATUS, * LPWFSPTRSTATUS;

typedef struct _wfs_ptr_caps
{
    WORD            wClass;
    WORD            fwType;
    BOOL            bCompound;
    WORD            wResolution;
    WORD            fwReadForm;
    WORD            fwWriteForm;
    WORD            fwExtents;
    WORD            fwControl;
    USHORT          usMaxMediaOnStacker;
    BOOL            bAcceptMedia;
    BOOL            bMultiPage;
    WORD            fwPaperSources;
    BOOL            bMediaTaken;
    USHORT          usRetractBins;
    LPUSHORT        lpusMaxRetract;
    WORD            fwImageType;
    WORD            fwFrontImageColorFormat;
    WORD            fwBackImageColorFormat;
    WORD            fwCodelineFormat;
    WORD            fwImageSource;
    WORD            fwCharSupport;
    BOOL            bDispensePaper;
    LPSTR           lpszExtra;
} WFSPTRCAPS, * LPWFSPTRCAPS;

typedef struct _wfs_frm_header
{
    LPSTR           lpszFormName;
    WORD            wBase;
    WORD            wUnitX;
    WORD            wUnitY;
    WORD            wWidth;
    WORD            wHeight;
    WORD            wAlignment;
    WORD            wOrientation;
    WORD            wOffsetX;
    WORD            wOffsetY;
    WORD            wVersionMajor;
    WORD            wVersionMinor;
    LPSTR           lpszUserPrompt;
    WORD            fwCharSupport;
    LPSTR           lpszFields;
} WFSFRMHEADER, * LPWFSFRMHEADER;

typedef struct _wfs_frm_media
{
    WORD            fwMediaType;
    WORD            wBase;
    WORD            wUnitX;
    WORD            wUnitY;
    WORD            wSizeWidth;
    WORD            wSizeHeight;
    WORD            wPageCount;
    WORD            wLineCount;
    WORD            wPrintAreaX;
    WORD            wPrintAreaY;
    WORD            wPrintAreaWidth;
    WORD            wPrintAreaHeight;
    WORD            wRestrictedAreaX;
    WORD            wRestrictedAreaY;
    WORD            wRestrictedAreaWidth;
    WORD            wRestrictedAreaHeight;
    WORD            wStagger;
    WORD            wFoldType;
    WORD            wPaperSources;
} WFSFRMMEDIA, * LPWFSFRMMEDIA;
```

```
typedef struct _wfs_ptr_query_field
{
    LPSTR           lpszFormName;
    LPSTR           lpszFieldName;
} WFSPTRQUERYFIELD, * LPWFSPTRQUERYFIELD;

typedef struct _wfs_frm_field
{
    LPSTR           lpszFieldName;
    WORD            wIndexCount;
    WORD            fwType;
    WORD            fwClass;
    WORD            fwAccess;
    WORD            fwOverflow;
    LPSTR           lpszInitialValue;
    LPWSTR          lpszUNICODEInitialValue;
    LPSTR           lpszFormat;
    LPWSTR          lpszUNICODEFormat;
} WFSFRMFIELD, * LPWFSFRMFIELD;


/*=================================================================*/
/* PTR Execute Command Structures */
/*=================================================================*/

typedef struct _wfs_ptr_print_form
{
    LPSTR           lpszFormName;
    LPSTR           lpszMediaName;
    WORD            wAlignment;
    WORD            wOffsetX;
    WORD            wOffsetY;
    WORD            wResolution;
    DWORD           dwMediaControl;
    LPSTR           lpszFields;
    LPWSTR          lpszUNICODEFields;
    WORD            wPaperSource;
} WFSPTRPRINTFORM, * LPWFSPTRPRINTFORM;

typedef struct _wfs_ptr_read_form
{
    LPSTR           lpszFormName;
    LPSTR           lpszFieldNames;
    LPSTR           lpszMediaName;
    DWORD           dwMediaControl;
} WFSPTRREADFORM, * LPWFSPTRREADFORM;

typedef struct _wfs_ptr_read_form_out
{
    LPSTR           lpszFields;
    LPWSTR          lpszUNICODEFields;
} WFSPTRREADFORMOUT, * LPWFSPTRREADFORMOUT;

typedef struct _wfs_ptr_raw_data
{
    WORD            wInputData;
    ULONG           ulSize;
    LPBYTE          lpbData;
} WFSPTRRAWDATA, * LPWFSPTRRAWDATA;

typedef struct _wfs_ptr_raw_data_in
{
    ULONG           ulSize;
    LPBYTE          lpbData;
} WFSPTRRAWDATAIN, * LPWFSPTRRAWDATAIN;

typedef struct _wfs_ptr_media_unit
{
    WORD            wBase;
    WORD            wUnitX;
    WORD            wUnitY;
} WFSPTRMEDIAUNIT, * LPWFSPTRMEDIAUNIT;

typedef struct _wfs_ptr_media_ext
```

```
{
    ULONG           ulSizeX;
    ULONG           ulSizeY;
} WFSPTRMEDIAEXT, * LPWFSPTRMEDIAEXT;


typedef struct _wfs_ptr_image_request
{
    WORD           wFrontImageType;
    WORD           wBackImageType;
    WORD           wFrontImageColorFormat;
    WORD           wBackImageColorFormat;
    WORD           wCodelineFormat;
    WORD           fwImageSource;
    LPSTR          lpszFrontImageFile;
    LPSTR          lpszBackImageFile;
} WFSPTRIMAGEREQUEST, * LPWFSPTRIMAGEREQUEST;


typedef struct _wfs_ptr_image
{
    WORD           wImageSource;
    WORD           wStatus;
    ULONG          ulDataLength;
    LPBYTE         lpbData;
} WFSPTRIMAGE, * LPWFSPTRIMAGE;

typedef struct _wfs_ptr_reset
{
    DWORD         dwMediaControl;
    USHORT        usRetractBinNumber;
} WFSPTRRESET, * LPWFSPTRRESET;


/*=================================================================*/
/* PTR Message Structures */
/*=================================================================*/

typedef struct _wfs_ptr_field_failure
{
    LPSTR           lpszFormName;
    LPSTR           lpszFieldName;
    WORD            wFailure;
} WFSPTRFIELDFAIL, * LPWFSPTRFIELDFAIL;

typedef struct _wfs_ptr_bin_threshold
{
    USHORT        usBinNumber;
    WORD          wRetractBin;
} WFSPTRBINTHRESHOLD, * LPWFSPTRBINTHRESHOLD;

typedef struct _wfs_ptr_paper_threshold
{
     WORD          wPaperSource;
     WORD          wPaperThreshold;
} WFSPTRPAPERTHRESHOLD, * LPWFSPTRPAPERTHRESHOLD;

typedef struct _wfs_ptr_media_detected
{
    WORD          wPosition;
    USHORT        usRetractBinNumber;
} WFSPTRMEDIADETECTED, * LPWFSPTRMEDIADETECTED;

/*   restore alignment   */
#pragma pack(pop)

#ifdef __cplusplus
}       /*extern "C"*/
#endif
#endif    /* __INC_XFSPTR__H */
```