New FileSpec and IPP Document Description attributes for version, etc.
From: Tom Hastings
Date: ~~January 21,~~ March 18,2003
File: New-FileSpec-and-IPP-attrs-030318.doc

This document contains proposed additions and clarifications to the JDF FileSpec resource for JDF/1.2 and the corresponding IPP Document Description attribute proposed for the IPP Document object extensions. The reason for showing both in the same document is to try to align the semantics where possible. Some of us believe that there will be significant interworking between JDF and other systems, such as IPP. Having the same values of corresponding attributes will make gateways a lot simpler. Note: There is no need for the names of the attributes to be the same. The real gain is for the values.

These proposals build on the proposals from Martin Bailey, Bob Taylor, and Israel Viente in CIP4 and the proposals from Bob Taylor, Dave Hall, and Peter Zehler in the PWG for inclusion in the IPP Document Object, PWG Semantic Model and PWG Print Services Interface (PSI) which is a follow on to Bluetooth.

JDF issues are highlighted in yellow like this.

JDF additions and clarifications are highlighted like this.

IPP issues are highlighted in green like this and are duplicates of the ones in the Document Object specification that the PWG Semantic Model WG is reviewing.

General ISSUE 01: for JDF what to do about enumerations that have already been defined in JDF/1.1, but haven't been using IANA registered values. See http://www.iana.org for all their registries, such as Operating System Names, MIME Types, etc. Should these values be tweaked to make them be spelled like the IANA registered values for JDF/1.2? See below for specific cases.

1 Summary of the additions to JDF and IPP

To align these two specs, add 4 attributes and 1 sub-element to the JDF/1.2 FileSpec resource spec:

FileClass attribute
FileFormatVersion attribute,
IEEE1284DeviceId attribute,
TransferEncoding
FileSpecParts sub-element

and clarify OSVersion.

To align these two specs, add 7 Document Description attributes (and their corresponding operation attributes) to the IPP Document object spec:

document-creator-application-name (name(MAX)),
document-creator-os-name (name(40)),
document-creator-application-version (text(127),
document-format-device-id (text(127)),
document-format-version (text(127)),
document-creator-os-version (text(40)),
document-container-summary (1setOf collection).

Comments and questions on this comparison and the IPP specification are in order.

2 Proposed additions to JDF FileSpec resource for JDF/1.2

Here is the JDF/1.1 FileSpec specification with the proposed additions and clarifications indicated using revision marks:

7.2.55 FileSpec

Modified in JDF 1.2

[Added 4 attributes: *FileClass*, *FileFormatVersion*, *IEEE1284DeviceId*, and *TransferEncoding*, added 1 sub-element: FileSpecParts, 9 ISSUES]

Specification of a file or a set of files.

Resource Properties

Resource class: Parameter
Resource referenced by: Error! Reference source not found.~~DBMergeParams~~, Error! Reference source not found.~~LayoutElement~~, Error! Reference source not found.~~PDLResourceAlias~~, Error! Reference source not found.~~ScanParams~~
Example Partition: *Separation*
Input of processes: -
Output of processes: -

Resource Structure

Name	Data Type	Description
Application ?	string	Creator application, such as Photoshop or Microsoft Word. See *AppVersion* for the version of the application.
AppOS ? Modified in JDF 1.2	enumeration	Operating system of the application that created the file. See *OSVersion* for the version number of the operating system. Possible values are: *Unknown* – Default value *DG_UX* *HP_UX* *IRIX* *Linux* *Mac* *Solaris* *Sun-OS* *VMS* *Windows* ~~*Linux*~~ ~~*Solaris*~~ ~~*IRIX*~~ ~~*DG_UX*~~ ~~*HP_UX*~~
AppVersion ?	string	Version of the value of the *Application* attribute.
CheckSum ? New in JDF 1.1 Modified in JDF 1.1A	hexBinary	Checksum of the file being referenced using the RSA MD5 algorithm. In JDF 1.1a, the term RSA MD was completed to RSA MD5. The data type was modified to hexBinary to accommodate the 128 bit output of the MD5 algorithm.

Name	Data Type	Description
Compression ?	enumeration	Indicates how the file is compressed. Possible values are: *None* – The file is not compressed. Default value. *Deflate* – The file is compressed using ZIP public domain compression (RFC 1951) *Gzip* – GNU zip compression technology (RFC 1952) *Compress* – UNIX compression (RFC 1977)
Disposition ?	enumeration	Indicates what the device should do with the file when the process that uses this resource as an input resource completes. Possible values are: *Unlink* – The device should release the file. *Delete* – The device should attempt to delete the file. *Retain* – The device should do nothing with the file. Default value.
DocumentNaturalLang ?	language	The natural language of the document this FileSpec refers to. Example values: 'en': for English 'en-us': for US English 'fr': for French 'de': for German
FileClass New in JDF 1.1 ISSUE 02: OK to add FileClass to JDF/1.2?	NMTOKEN	Identifies the class of file. Possible values include: *ByteMap* – the file is a byte map. See Byte Map resource *Font* – the file is a font. The FontDetails attribute indicates information about the Font. *Form* – the file is an electronic form that will be merged with other PDL data. *Logo* – the file is an electronic logo that will be merged with other PDL data. *Printable* – the file is printable. The *MIMEType* attribute must be supplied, unless the file format does not have either an IANA-registered MIME type or a private MIME type. The default.
FileFormat ?	string	A formatting string used with the *Template* attribute to define a sequence of filenames in a batch process. If neither *URL* nor *UID* are present, both *FileFormat* and *FileTemplate* must be present, unless the resource is a pipe. For more information, see the text following this table.

Name	Data Type	Description
FileFormatVersion ? New in JDF 1.2	string	The level, version, or standard's body designation of the file format identified by *MimeType*. Some possible values are the same as the Printer MIB [RFC1759] prtInterpreterLangLevel. Possible values from the Printer MIB include: "3" – level 3 of PostScript "5e" – version 5e of PCL 5e For those document formats that are defined in standards, other possible values are the official designation of that standard. Such values must use the standards body's complete designation and preserving case, SPACE characters, version numbers, dates (if used by the standards organization to designate the standard), and all special characters that are in the standard designation. These possible values include: "2000" – For Microsoft Word 2000 "ISO 12639-1998" – For ISO 12639:1998 standard that specifies TIFF/IT Profile 1 [ISO 12639]. "ISO 15930-1:2001~~PDF/X-1:2001~~" – For the ISO standard that specifies PDF/X [iso15930]. ISSUE 03: OK to use the official ISO standard number, part number and date: ISO 15930-1:2001" instead of "PDF/X-1:2001"? ISSUE 04: If so, how do you specify PDF/X-1a which is also specified in Part 1?
FileSize ?	integer	Size of the file in Byte.
FileTemplate ?	string	A template, used with *FileFormat*, to define a sequence of filenames in a batch process. If neither *URL* nor *UID* is present, both *FileFormat* and *FileTemplate* must be present, unless the resource is a pipe.
FileVersion ? New in JDF 1.1	string	Version of the file referenced by this FileSpec. ISSUE 05: Is *FileVersion* for file systems that have automatic version numbering, like Tenex and CVS?
IEEE1284DeviceId ? New in JDF 1.2	string	Identifies the type of device for which the document was formatted, including manufacturer and model. The value of this variable MUST exactly match the IEEE 1284-2000 Device ID string, except the length field MUST NOT be specified. See the Microsoft Universal Plug and Play [upnp] section 2.2.6 DeviceId parameter for details. Here is an example showing only the required fields for a PostScript document: `MANUFACTURER:ACME Co.;COMMAND SET:PS;MODEL:LaserBeam 9;` *(See IEEE 1284-2000 clause 7.6)*

Name	Data Type	Description
MimeType ? Modified in JDF 1.2	string	Mime type of the file. If the file format doesn't have either a MIME type registered with IANA [iana-mt] or a private MIME type, *FileClass* and/or *FileFormatVersion* may be supplied to identify the format of the file. Possible *MimeType* values include: ~~ISSUE (Martin): Need the names and references for these MIME types.~~ "application/msword" – Microsoft Word [iana-mt] "application/pdf" – Adobe Portable Document Format [iana-mt] "application/postscript" – Adobe PostScript™ See [rfc2045, rfc2046, iana-mt] "application/vnd.hp-PCL" – Hewlett Packard Printer Control Language (PCL™) [iana-mt] "application/zip" – public domain ZIP compression [rfc1951]. See also FileSpecParts element to summarize the contents. [iana-mt] "image/jpeg" – JPEG [rfc2045, rfc2046, iana-mt]] "image/tiff" – Tag Image File Format [rfc2302] "multipart/related" – related files combined as a MIME multipart package [rfc2387]. See also FileSpecParts element to summarize the contents. ISSUE 06: Which additional MIME type values from the IPP list should we add to JDF? ISSUE 07: What about adding "application/octet-stream" which means auto sense?
OSVersion ? Modified in JDF 1.2	string	Version of the operating system identified by *AppOS*. Possible values include: "Unknown" and the version number part of the IANA Registry of Operating System Names [iana-mt], not including the first HYPHEN (-) character that separates the name from the version. See: http://www.iana.org/assignments/operating-system-names. Examples of *AppOS* and *OSVersion* values: For *AppOS = "Linux": "2.2."* For *AppOS = "Mac": "X"* For *AppOS = "Sun-OS": "4.0"* For *AppOS* = "UNIX": "BSD", "V", "V.1", "V.2", "V.3", "PC". For *AppOS = "Windows": "95", "98", "NT", "NT-5", "2000", "XP".*
PageOrder ?	enumeration	Indicates whether the pages in the file are in reverse order. Possible values are: *Ascending* – The first page in the file is the lowest numbered page. *Descending* – The first page in the file is the highest numbered page.
ResourceUsage ?	NMTOKEN	If an element uses more than one FileSpec subelement, this attribute is used to refer from the parent element to a certain child element of this type, for example, see **Error! Reference source not found.**~~ColorSpaceConversionParams~~.

Name	Data Type	Description
TransferEncoding	NMTOKEN	Type of transfer encoding for purposes of transferring the files. *TransferEncoding* does not specify the character encoding of the files themselves. When receiving files, the receiver first decodes and applies the attributes in this order: (1) *TransferEncoding* (2) *MimeType* (3) if *MimeType* is a container type (e.g., multipart/related or application/zip), the *MimeType* value(s) in the FileSpecParts subelement(s) Possible values include: *Base64* - A format for encoding arbitrary binary information for transmission by electronic mail. *BinHex* - Binhex encoding converts an 8-bit file into a 7-bit format, similar to uuencoding. *Binhex* format preserves file attributes, as well as Macintosh resource forks, and includes CRC (Cyclic Redundancy Check) error-checking. This encoding method works on any type of file, including formatted word processing and spreadsheet files, graphics files, and even executable files (i.e. programs or applications). Encoded files usually have a .HQX extension. Note: *BinHex* is not to be confused with *MacBinary* encoding, which is an 8-bit format. For more information see: http://www.natural-innovations.com/boo/binhex.html#info *MacBinary* -A format that combines the two forks of a Mac file, together with the file information (Name, Creator Application, File Type, etc) into a single binary data stream, suitable for storage or transferring through non-Mac systems. For more information see: http://astronomy.swin.edu.au/~pbourke/dataformats/macbinary/ *None* - Default *UUEncode* - A set of algorithms for converting files into a series of 7-bit ASCII characters that can be transmitted over the Internet. Originally, uuencode stood for Unix-to-Unix encode, but it has since become a universal protocol used to transfer files between different platforms such as Unix, Windows, and Macintosh. Uuencoding is especially popular for sending e-mail attachments. For more information see: http://www.webopedia.com/TERM/U/Uuencode.html
UID ? New in JDF 1.1	string	Unique internal ID of the referenced file. This attribute is dependent on the type of file that is referenced: PDF: Variable unique identifier in the ID field of the PDF file's trailer. ICC Profile: Profile ID in byte 84-99 of the ICC profile header. Others – Format specific.
URL ?	URL	Location of the file. If *URL* is not present, and neither *FileFormat* nor *FileTemplate* are present, the referencing resource must be a pipe.
UserFileName ?	string	A user-friendly name which may be used to identify the file, but is not guaranteed to be unique.

Name	Data Type	Description
FileAlias *	element	Defines a set of mappings between file names that may occur in the document and URLs (which may refer to external files or parts of a MIME message). ISSUE 08: I don't understand what a FileAlias is. Can we have an example and a better explanation?
FileSpecParts * New in JDF 1.2	element	When MimeType is a container file format, such as "application/zip' or "multipart/related" [RFC2387], this subelement summarizes the distinct types of files in the container file. The purpose of the FileSpecParts element is to allow a receiving Printer to determine whether or not it supports all of the file formats in the supplied instance of the container. However, the FileSpecParts element does not provide means to associate each element instance with a particular file in the container file, so FileSpecParts does not provide a "manifest" of the container. There must note be any duplicate FileSpecParts elements values, that is, no elements with all the same attribute values; its a set, not a sequence. So 1 PCL file and 100 PostScript files with the same details in a .zip file would have MimeType = "application/zip" at the top level and 2 FileSpecParts sub-elements: one with MimeType = "application/vnd.hp-PCL" and the other with MimeType = "application/postscript" If a file in a container file is itself a container file, the single FileSpecParts element(s) SHOULD contain the flattened distinct collection values for all files at all nested levels. The FileSpecParts element(s) is *not* recursively defined to contain further FileSpecParts element(s).

Structure of FileAlias Subelement

Name	Data Type	Description
Alias	string	The filename which is expected to occur in the file.
Disposition	enumeration	Indicates what the device should do with the file referenced by this alias when the process that uses this resource as an input resource completes. Possible values are: *Unlink* – The device should release the file. *Delete* – The device should attempt to delete the file. *Retain* – The device should do nothing with the file.
MimeType ?	string	Mime type of the file.
URL	URL	The URL which identifies the file the alias refers to.

Structure of FileSpecParts Subelement

Name	Data Type	Description
Application ?	string	See FileSpec::*Application*.
AppOS ?	enumeration	See FileSpec::*AppOS*.
AppVersion ?	string	See FileSpec::*AppVersion*.
DocumentNaturalLang ?	language	See FileSpec::*DocumentNaturalLang*.
FileFormatVersion ?	string	See FileSpec::*FileFormatVersion*.
FileTemplate ?	string	See FileSpec::*FileTemplate*. ISSUE 09: Should FileTemplate be included in FileSpecParts or not?

IEEE1284DeviceId ?	string	See FileSpec::*IEEE1284DeviceId*
MimeType ?	string	See FileSpec::*MimeType*
OSVersion ?	string	See FileSpec::*OSVersion*

Usage of Format and Template

The function defined when using the attributes *FileFormat* and *FileTemplate* is drawn from the same root as the standard C print function and, therefore, overtly resembles the model of that function. *FileFormat* is the first argument and *FileTemplate* is a comma-separated list of the additional arguments. *FileTemplate* may contain the following operators : +,-,*,/,%,(,) which are evaluated using standard C-operator precedence and the variables defined in the following table:

Table 2~~7~~-1~~86~~ Predefined variables used in FileTemplate

Name	Description
element	Integer iterator over all elements in a given page. Restarts at 0 for each page.
i	Integer iterator over all files produced by this process. 0-based numbering.
page	Integer iterator over the page number of a document. This is equivalent to r for the case that each run contains exactly one page.
r	Integer iterator over all RunList partitions with a partition key of "*Run*" in an input **RunList**.
ri	Integer iterator over all indices in an input Run of a **RunList**. This index is equivalent to looping over a RunIndex.
sep	Separation as defined in the separation PartIDKey of a partitioned resource.
surf	Surface string, "Front" or "Back"
SheetName	SheetName string of a partitioned resource.
SignatureName	SignatureName string of a partitioned resource.
TileX	X coordinate of a Tile
TileY	Y coordinate of a Tile
PartVersion	PartVersion string of a partitioned resource.
jobPartID	JobPartID string
jobID	Job ID string
jobName	*DescriptiveName* of the Node that is being processed.
Time	Current *Time* in ISO 8601 format.
Date	Current *Date* in ISO 8601 format.
CustomerID	CustomerID

Example:

```
<FileSpec FileFormat = "file://here/next/%s/%4.i/m%4.i.pdf" FileTemplate =
"JobID,i/100,i%100"/>
```

with JobID = "j001" and a **RunList** defining 2023 created files will iterate all created files and place them into:

```
"file://here/next/j001/0000/m0000.pdf"
…
"file://here/next/j001/0020/m0023.pdf"
```

3 Additional references to add to both JDF/1.2 and IPP Document Object specs

Add the following References to JDF/1.2 References section and use these symbolic tags to refer to them throughout the spec:

[iana-mt] -

[iso12639] - ISO 12639:1998 Graphic technology -- Prepress digital data exchange -- Tag image file format for image technology (TIFF/IT).

[iso15930] - ISO 15930-1:2001 Graphic technology -- Prepress digital data exchange -- Use of PDF -- Part 1: Complete exchange using CMYK data (PDF/X-1 and PDF/X-1a).

[rfc1759] - Smith, R., Wright, F., Hastings, T., Zilles, S., and Gyllenskog, J., "Printer MIB", RFC 1759, March 1995.

[rfc1951] -

[rfc2045] -

[rfc2046] -

[rfc2302] -

[rfc2387] - E. Levinson, “The MIME Multipart/Related Content-type”, RFC 2387, August 1998.

4 Comparison of JDF/1.1 FileSpec and IPP/1.1 Document Description attributes

This section is a comparison of the JDF FileSpec resource including additions for JDF/1.2 and the proposed IPP Document object attributes. The IPP semantics are taken from [RFC2911] and the IPP Document object specification, version 0.7, 3/14/03. See ftp://ftp.pwg.org/pub/pwg/ipp/new_DOC/wd-ippdoc10-20030314-doc.zip
The Document Description attributes are proposed to be added to IPP to meet the requirements of a number of print protocols to have additional information about a Document Format, than just its MIME type. So this attribute has additional fields (member attributes) for version, natural language, platform (OS on which the document was generated), and device ID. It also caters to MIME Media Types that are containers, such as application/zip and multipart/related, where contain additional document of possibly differing MIME media types.

It would be good to add the same attributes to JDF, presumably to the FileSpec resource. Also JDF has a number of attributes which would be good to add to the IPP Document object.

The ISSUES for JDF are a repeat of the ones above and so are not numbered.

Here is a comparison of JDF/1.1 FileSpec and IPP/1.1 and IPP “document-format-detail” attribute:

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
Application (string) Creator application, such as Photoshop or Microsoft Word. See *AppVersion* for the version of the application.	document-creator-application-name (name(MAX)) This OPTIONAL Document Description identifies the application that created the document. The version number MUST NOT be included (see the "document-creator-application-version" attribute). Examples:	Add the “document-creator-application-name” attribute to IPP
	“Photoshop”, “Microsoft Word”.	
AppOS (enumerations) Modified in JDF 1.2 Operating system of the application that created the file. See *OSVersion* for the version number of the operating system. Possible values are:	document-creator-os-name (name(40)) This OPTIONAL Document Description attribute identifies the name of the operating system on which the document was generated. Valid values are the operating system names defined in the IANA document [os-names] with the version number portion removed (see the “document-creator-os-version” attribute)IANA Operating System Names are consist of up to 40 uppercase US-ASCII letters, hyphen (“-“), period (“.”), and slash (“/”) characters. The zero length string represents unknown (rather than the	Same semantics, but JDF doesn’t use the IANA OS Name Registry, so JDF has some values that represent operating systems that aren’t registered with IANA. Clarify JDF *AppOS*

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
	UNKNOWN value in the IANA OS Registry, since clients are not expected to localize names). Example IANA OS Registry values:	
Unknown – Default value *DG_UX* *HP_UX* *IRIX* *Linux* *Mac* *Solaris* *Windows* *ISSUE: Align these values with the IANA and IPP ? See next column*	‘AIX, ‘DOS’, ‘LINUX’, ‘MACOS’, ‘MSDOS’, ‘MVS’, ‘NETWARE’, ‘OS/2’, ‘SUN’, ‘SUN-OS’, ‘UNIX’, ‘VMS’, ‘WINDOWS’.	ISSUE 11: JDF uses its own enumerations, instead of the IANA OS name registry. Also need to sort JDF values alphabetically.
AppVersion (string) Version of the value of the *Application* attribute.	document-creator-application-version (text(127)) The OPTIONAL Document Description attribute identifies the version number of the application that created the document. The version number MUST NOT include the application name. See “document-creator-application-name” attribute. Examples:	Add "document-creator-application-version" to IPP
	'V3.0.', ‘V6.0’	
*CheckSum (*hexBinary) New in JDF 1.1 Modified in JDF 1.1A Checksum of the file being referenced using the RSA MD5 algorithm. In JDF 1.1a, the term RSA MD was completed to RSA MD5. The data type was modified to hexBinary to accommodate the 128 bit output of the MD5 algorithm.	-	Handled in IPP by the TLS lower layer security.
Compression (enumeration) Indicates how the file is compressed. Possible values are:	“compression” (type3 keyword) Identifies the set of supported compression algorithms for document data. Compression only applies to the document data; compression does not apply to the encoding of the IPP operation itself. Standard keyword values are:	Same semantics and values (except for case).
None – The file is not compressed. Default value. *Deflate* – The file is compressed using ZIP public domain compression (RFC 1951)	'none': no compression is used. 'deflate': ZIP public domain inflate/deflate) compression technology in RFC 1951 [RFC1951]	

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
Gzip – GNU zip compression technology (RFC 1952) *Compress* – UNIX compression (RFC 1977)	'gzip' GNU zip compression technology described in RFC 1952 [RFC1952]. 'compress': UNIX compression technology in RFC 1977 [RFC1977]	
*Disposition (*enumeration) Indicates what the device should do with the file when the process that uses this resource as an input resource completes. Possible values are:	-	ISSUE12: I think LPR has this, right? So should we add “document-disposition” to IPP?
Unlink – The device should release the file. *Delete* – The device should attempt to delete the file. *Retain* – The device should do nothing with the file. Default value.		
*DocumentNaturalLang (*language) The natural language of the document this FileSpec refers to. Example values: 'en': for English 'en-us': for US English 'fr': for French 'de': for German	“document-natural-language” (naturalLanguage) This OPTIONAL Document Description attribute specified the natural language of the document (see [rfc2911] §3.2.1.1 and [pwg5100.4] §5.1.7). The Printer sets this Document Description attribute from the corresponding operation attribute supplied by the client in the Document Creation operation (see section **Error! Reference source not found.**~~3.1~~). The Printer MAY use this value to select fonts or other Globalization processing. Examples include: ISSUE: The definition in [rfc2911] §3.2.1.1 and [pwg5100.4] §5.1.7 is single-valued. OK that this document attribute isn’t 1setOf? ISSUE: Or should we extend “document-natural-language” to 1setOf naturalLanguage) and keep the same name? ISSUE: Or change the name to “document-natural-languages”?	Same semantics and values
	'en': for English 'en-us': for US English 'fr': for French 'de': for German	
FileClass (NMTOKEN) New in JDF 1.1 ISSUE: OK to add FileClass to JDF/1.2? Identifies the class of file. Possible values include:	-	Add a similar attribute to IPP when requested. For IPP need to think about whether to introduce a generic Resource object for which font could be

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
		object for which font could be a specialization or whether to introduce each object class as its own object with its own attributes and operations.
ByteMap – the file is a byte map. See Byte Map resource *Font* – the file is a font. The FontDetails attribute indicates information about the Font. *Form* – the file is an electronic form that will be merged with other PDL data. *Logo* – the file is an electronic logo that will be merged with other PDL data. *Printable* – the file is printable. The *MIMEType* attribute must be supplied, unless the file format does not have either an IANA-registered MIME type or a private MIME type. The default.		
FileFormat (string) A formatting string used with the *Template* attribute to define a sequence of filenames in a batch process. If neither *URL* nor *UID* are present, both *FileFormat* and *FileTemplate* must be present, unless the resource is a pipe. For more information, see the text following this table.	-	Not sure I understand. Don't put in IPP.
FileFormatVersion ? (string) The level, version, or standard's body designation of the file format identified by *MimeType*. Some possible values are the same as the Printer MIB [RFC1759] prtInterpreterLangLevel. Possible values from the Printer MIB include: "3" – level 3 of PostScript "5e" – version 5e of PCL 5e For those document formats that are defined in standards, other possible values are the official designation of that standard. Such values must use the standards body's complete designation and preserving case, SPACE characters, version numbers, dates (if used by the standards organization to designate the standard), and all special characters that are in the standard designation. These possible values include:	document-format-version (text(127)) This REQUIRED Document Description attribute contains the level or version of the document format. Possible values are the same as the Printer MIB [rfc1759] prtInterpreterLangLevel (not prtInterpreterLangVersion). For those document formats that are defined in standards, the "document-format-version" may contain the official designation of that standard. ISSUE 10: OK that "document-format-version" is REQUIRED for a Printer to support? ISSUE: 11: The problem with separating "document-format" and "document-format-version" is how can a Printer describe what versions are supported, since the versions have to be associated with the document format. Standard text values are:	Add FileFormatVersion to JDF.

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
"2000" – For Microsoft Word 2000 "ISO 12639-1998" – For ISO 12639:1998 standard that specifies TIFF/IT Profile 1 [ISO 12639]. "ISO 15930-1:2001~~PDF/X-1:2001~~" – For the ISO standard that specifies PDF/X [iso15930]. ISSUE: OK to use the official ISO standard designation, rather than "PDF/X-1-2001"? ISSUE: If so, how do you specify PDF/X-1a which is also specified in Part 1? ~~ISSUE: Bob Taylor and Martin Bailey think that this name is confusing, since MIME Types don't have versions, but the document formats that they represent do. Call it FileFormatVersion instead?~~	'2000': For MS-WORD 2000. '3': For Postscript level 3 [rfc1759]. '5e': For PCL 5e [rfc1759]. 'ISO 12639~~-1~~:1998~~6~~': For ISO 12639:1998~~6~~ standard that specifies TIFF/IT Profile 1 [ISO 12639] '**PDF**/**X**-1:2001': For the ISO standard that specifies PDF/X [iso15930]. ISSUE 12: Or should the official ISO standard number, part number and date, be used instead, e.g., "ISO 15930-1:2001"? ISSUE: If so, how do you specify PDF/X-1a which is also specified in Part 1?	
FileSize (integer) Size of the file in Byte.	"k-octets" (integer(0:MAX)) This OPTIONAL "k-octets" Document Description attribute has the same semantics as the corresponding "job-k-octets" Job Description attribute (see [rfc2911] §4.3.17.1) applied to the Document object. The Printer sets this Document Description attribute from the corresponding operation attribute supplied by the client in the Document Creation operation (see section **Error! Reference source not found.**~~3.1~~). "job-k-octets" (integer(0:MAX)) This attribute specifies the total size of the document(s) in K octets, i.e., in units of 1024 octets requested to be processed in the job. The value MUST be rounded up, so that a job between 1 and 1024 octets MUST be indicated as being 1, 1025 to 2048 MUST be 2, etc.	Same semantics, different units of measure.
FileTemplate (string) A template, used with *FileFormat*, to define a sequence of filenames in a batch process. If neither *URL* nor *UID* is present, both *FileFormat* and *FileTemplate* must be present, unless the resource is a pipe.	-	Not sure I understand. Don't put in IPP.
FileVersion (string) New in JDF 1.1 Version of the file referenced by this FileSpec.	-	Not sure I understand. Don't put in IPP.

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
ISSUE: Is this for file systems that have automatic version numbering, like Tenex and CVS?		
IEEE1284DeviceId ? (string) ISSUE: OK, to add *IEEE1284DeviceId* to JDF/1.2? Identifies the type of device for which the document was formatted, including manufacturer and model. The value of this variable MUST exactly match the IEEE 1284-2000 Device ID string, except the length field MUST NOT be specified. See the Microsoft Universal Plug and Play [upnp] section 2.2.6 DeviceId parameter for details. Here is an example showing only the required fields for a PostScript document:	document-format-device-id (text(127)) This OPTIONAL Document Description attribute identifies the type of device for which the document was formatted, including manufacturer and model. This attribute is intended to identify document formats that are not portable, e.g., PDLs that are device dependent. The value of this variable MUST exactly match the IEEE 1284-2000 Device ID string (see [IEEE1284] clause 6), except the length field MUST NOT be specified. See the Microsoft Universal Plug and Play [upnp] section 2.2.6 DeviceId parameter for details and examples. Here is an example showing only the required fields for a PostScript document:	Add IEEE1284DeviceId to JDF FileSpec resource
`MANUFACTURER:ACME Co.;COMMAND SET:PS;MODEL:LaserBeam 9;` *(See IEEE 1284-2000 clause 7.6)*	MANUFACTURER:ACME Co.;COMMAND SET:PS;MODEL:LaserBeam 9;	
MimeType (string) Mime type of the file. If the file format doesn't have either a MIME type registered with IANA [iana-mt] or a private MIME type, *FileClass* and/or *FileFormatVersion* may be supplied to identify the format of the file. Possible *MimeType* values include: ~~ISSUE (Martin): Need the names and references for these MIME types.~~	document-format (mimeMediaType) This REQUIRED Document Description attribute specifies the document format (see [rfc2911] §3.2.1.1) for the Document object. The standard values for this attribute are Internet Media types (sometimes called MIME types). For further details see the description of the 'mimeMediaType' attribute syntax in [rfc2911] section 4.1.9. If it is a MIME Media Type, such as 'multipart/related' or 'application/zip', that is a container format that contains document parts, the "document-container-summary" attribute summarizes the content (see section **Error! Reference source not found.**~~8.2.8~~). The Printer sets this Document Description attribute from the corresponding operation attribute supplied by the client in the Document Creation operation (see section **Error! Reference source not found.**~~3.1~~). Example values:	Same semantics and values
"application/msword" – Microsoft Word [iana-mt] "application/pdf" – Adobe Portable Document Format [iana-mt] "application/postscript" – Adobe PostScript™ See [RFC2045,RFC2046]	'text/html': An HTML document 'text/plain': A plain text document in US-ASCII (RFC 2046 indicates that in the absence of the charset parameter MUST mean US-ASCII rather than simply unspecified) [RFC2046]. 'text/plain; charset=US-ASCII': A plain text document in US-ASCII	

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
"application/vnd.hp-PCL" – Hewlett Packard Printer Control Language (PCL™) [iana-mt] "application/zip" – public domain ZIP compression [rfc1951]. See also FileSpecParts element to summarize the contents. "image/jpeg" – JPEG [rfc2045, rfc2046] "image/tiff" – Tag Image File Format [rfc2302] "multipart/related" – related files combined as a MIME multipart package [rfc2387]. See also FileSpecParts element to summarize the contents. ISSUE: Which additional values from the IPP list should we add to JDF? ISSUE: What about adding "application/octet-stream" which means auto sense?	[RFC2046]. 'text/plain; charset=ISO-8859-1': A plain text document in ISO 8859-1 (Latin 1) [ISO8859-1]. 'text/plain; charset=utf-8': A plain text document in ISO 10646 represented as UTF-8 [RFC2279] 'application/postscript': A PostScript document [RFC2046] 'application/vnd.hp-PCL': A PCL document [IANA-MT] (charset escape sequence embedded in the document data) 'application/pdf': Portable Document Format - see IANA MIME Media Type registry 'application/octet-stream': Auto-sense - see [rfc2911] section 4.1.9.1 'application/zip': ZIP container file package [rfc1951]. See also "document-container-summary" (1setOf collection). to summarize the contents. 'multipart/related': related files combined as a MIME multipart package [rfc2387]. See also "document-container-summary" (1setOf collection). to summarize the contents.	
OSVersion (string) Version of the operating system identified by *AppOS*. identified by *AppOS*. Possible values include: "Unknown" and the version number part of the IANA Registry of Operating System Names [iana-mt], not including the first HYPHEN (-) character that separates the name from the version. See: http://www.iana.org/assignments/operating-system-names. Examples of *AppOS* and *OSVersion* values:	document-creator-os-version (text(40)) This OPTIONAL Document Description attribute identifies the version of the operating system on which the document was generated. Valid values include the version portion of any of the operating system names defined in the IANA Registry [os-names]. The value MUST NOT include the name portion of the registered OS name (see "document-creator-os-name" attribute). The zero length string represents unknown (rather than the UNKNOWN value in the IANA OS Registry, since clients are not expected to localize names). Example values for the indicated "document-creator-os-name" value:	ISSUE: Clarify JDF *OSVersion* . Note: while JDF doesn't reference the IANA Registry, the version numbers will tend to be the same or a superset of the IANA registry. Add "document-creator-os-version" attribute to IPP.
For *AppOS = "Linux": "2.2."* For *AppOS = "Mac": "X"* For *AppOS = "Sun-OS": "4.0"* For *AppOS* = "UNIX": "BSD", "V", "V.1", "V.2", "V.3", "PC". For *AppOS = "Windows": "95", "98", "NT", "NT-5",*	For 'AIX': '370', 'PS/2' For 'LINUX': '1.0', '1.2', '2,0', '2.2', '2.4' For 'MVS': 'SP' For 'NETWARE': '3', '3.11', '4.0', '4.1', '5.0' For 'SUN-OS': '3.5', '4.0'	

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
“2000”, “XP”.	For ‘UNIX’: ‘BSD’, ‘V’, ‘V.1’, ‘V.2’, ‘V.3’, ‘PC’ For ‘WINDOWS’: ‘95’, ‘98’, ‘CE’, ‘NT’, ‘NT-2’, ‘NT-3’, ‘NT-3.5’, ‘NT-3.51’, ‘NT-4’, ‘NT-5’, ‘2000’, ‘XP’ [not registered yet]	
PageOrder (enumeration) Indicates whether the pages in the file are in reverse order. Possible values are:	"page-order-received" (type2 keyword) This attribute specifies the page order of the print-stream pages defined in the document data. The "page-order-received" attribute does not provide any direct processing instructions, it only provides information about the page order so that the client can specify ordinal page numbers with respect to the original source document, rather than having to take into account whether the print stream pages are being sent "one to N" or "N to one". For example, consider such Job Template attributes as "insert-sheet" ([pwg5100.3] section 3.5) and "page-overrides" (see [ipp-override]). See [pwg5100.3] section 2.5 for a complete discussion of print-stream page order. Standard keyword values are:	Same semantics
Ascending – The first page in the file is the lowest numbered page. *Descending* – The first page in the file is the highest numbered page.	'1-to-n-order' 'n-to-1-order'	Mappable values
ResourceUsage (NMTOKEN) If an element uses more than one FileSpec subelement, this attribute is used to refer from the parent element to a certain child element of this type, for example, see **Error! Reference source not found.**~~ColorSpaceConversionParams~~.	-	Don't put in IPP.
TransferEncoding (NMTOKEN) Type of transfer encoding for purposes of transferring the files. *TransferEncoding* does not specify the character encoding of the files themselves. When receiving files, the receiver first decodes and applies the attributes in this order: (1) *TransferEncoding* (2) *MimeType* (3) if *MimeType* is a container type (e.g., multipart/related or application/zip), the *MimeType in the* FileSpecParts subelement Possible values include:	-	Don’t add to IPP now, but add “document-transfer-encoding” to IPP in the future when needed.

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
Base64 - A format for encoding arbitrary binary information for transmission by electronic mail. *BinHex* - Binhex encoding converts an 8-bit file into a 7-bit format, similar to uuencoding. *Binhex* format preserves file attributes, as well as Macintosh resource forks, and includes CRC (Cyclic Redundancy Check) error-checking. This encoding method works on any type of file, including formatted word processing and spreadsheet files, graphics files, and even executable files (i.e. programs or applications). Encoded files usually have a .HQX extension. Note: *BinHex* is not to be confused with *MacBinary* encoding, which is an 8-bit format. For more information see: http://www.natural-innovations.com/boo/binhex.html#info *MacBinary* -A format that combines the two forks of a Mac file, together with the file information (Name, Creator Application, File Type, etc) into a single binary data stream, suitable for storage or transferring through non-Mac systems. For more information see: http://astronomy.swin.edu.au/~pbourke/dataformats/macbinary/ *None* - Default *UUEncode* - A set of algorithms for converting files into a series of 7-bit ASCII characters that can be transmitted over the Internet. Originally, uuencode stood for Unix-to-Unix encode, but it has since become a universal protocol used to transfer files between different platforms such as Unix, Windows, and Macintosh. Uuencoding is especially popular for sending e-mail attachments. For more information see: http://www.webopedia.com/TERM/U/Uuencode.html		
UID (string) Unique internal ID of the referenced file. This attribute is dependent on the type of file that is referenced: PDF: Variable unique identifier in the ID field of the PDF file’s	-	Don't put in IPP.

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
trailer. ICC Profile: Profile ID in byte 84-99 of the ICC profile header. Others – Format specific.		
URL (string) Location of the file. If *URL* is not present, and neither *FileFormat* nor *FileTemplate* are present, the referencing resource must be a pipe.	"document-uri" (uri) The URI of the document to be printed by reference. The Printer pulls the file at its convenience.	Same semantics and values.
UserFileName (string) A user-friendly name which may be used to identify the file, but is not guaranteed to be unique.	-	Clarify JDF *UserFileName* Don't add "user-file-name" to IPP until requested.
FileAlias *(element) Defines a set of mappings between file names that may occur in the document and URLs (which may refer to external files or parts of a MIME message). ISSUE: I don't understand what a FileAlias is. Can we have an example and a better explanation?	-	Not sure I understand. Don't put in IPP.
FileSpecParts * (subelement) When MimeType is a container file format, such as "application/zip' or "multipart/related" [RFC2387], this subelement summarizes the distinct types of files in the container file. The purpose of the FileSpecParts element is to allow a receiving Printer to determine whether or not it supports all of the file formats in the supplied instance of the container. However, the FileSpecParts element does not provide means to associate each element instance with a particular file in the container file, so FileSpecParts does not provide a "manifest" of the container. There must note be any duplicate FileSpecParts elements values, that is, no elements with all the same attribute values; its a set, not a sequence. So 1 PCL file and 100 PostScript files with the same details in a .zip file would have MimeType = "application/zip" at the top level and 2 FileSpecParts sub-elements: one with MimeType = "application/vnd.hp-PCL" and the other with MimeType = "application/postscript" If a file in a container file is itself a container file, the single FileSpecParts element(s) SHOULD contain the flattened distinct collection values for all files at all nested levels. The	document-container-summary (1setOf collection) This OPTIONAL Document Description attribute summarizes the document format content of the body parts, if the document's document format is a container type, such as 'multipart/related' or 'application/zip'. If a Printer supports such a container MIME type, the Printer MUST support this "document-container-summary" Document Description attribute and all the member attributes in **Error! Reference source not found.**~~Table 10~~ that the Printer supports as top level Document Description attributes. The purpose of the "document-container-summary" attribute is to allow a receiving Printer to determine whether or not it supports all of the document formats in the supplied instance of the container. However, the "document-container-summary" attribute does not provide means to associate each collection value with particular document in the archive file, so it does not provide a "manifest" of the container. The member attributes defined for this collection are listed in [doc-obj] Table 10 and are the same as those defined for the corresponding Document Description attributes themselves, i.e., a recursive definition. But there MUST NOT be any duplicate collection values, that is no collection values with all the same	Add a DocumentPart to JDF FileSpec which has a subset of the FileSpec attributes. Include only the intensive attributes, i.e., the ones that can apply to multiple parts. ISSUE: Rainer suggest that the sub-parts of a zip file or multipart file can be represented using JDF partitioning, so that we don't need to add sub-elements to FileSpec.

JDF FileSpec attributes and elements	IPP Document Description attributes	Comments
distinct collection values for all files at all nested levels. The FileSpecParts element(s) is *not* recursively defined to contain further FileSpecParts element(s). So FileSpecParts includes only the following attributes: *Application ?* *AppOS ?* *AppVersion ?* *DocumentNaturalLang ?* *FileFormatVersion ?* *IEEE1284DeviceId ?* *MimeType* *OSVersion ?* Only include separate FileSpecParts instances which have different combinations of attributes and values. So if a .zip file contains 1 PCL file and 100 PostScript files all with the same attributes, only 2 DocumentPart subelements are supplied. ISSUE: Or should we just recursively refer to FileSpec and rule out more than one level?	member attribute values; its a set, not a sequence. So 100 PostScript files with the same details in a .zip file would have 'application/zip' as the MIME type for the top level "document-format" Document Description attribute for the Document object and a "document-container-summary" Document Description collection attribute with only one collection value containing a "document-format" member attribute with the 'application/postscript' MIME type value. If a file in a container file is itself a container file, the single "document-container-summary" (1setOf collection) Document Description attribute SHOULD contain the flattened distinct collection values for all files at all nested levels. The "document-container-summary" attribute is not recursively defined to contain "document-format-details". ISSUE 06: OK that "document-container-summary" is only one level deep? ISSUE 07: Is the description of "document-container-summary" attribute OK?	