

1 Job Monitoring MIB, V0.9~~10~~  
2 (This cover page is not part of the Internet-Draft  
3 that is being forwarded to the IESG to be an Informational RFC)  
4

5 From: Tom Hastings

6 Date: ~~01/13/98~~02/06/98

7 Version: 0.9~~10~~ (already numbered 1.0 in body, ~~waiting for proof~~  
8 ~~reading~~)

9 File: ftp://ftp.pwg.org/pub/jmp/mibs/jmp-mib.doc .pdf jmp-mib-  
10 ~~rev.doc .pdf .pdr~~

11 Status: ~~This specification was approved by Printer Working Group (PWG)~~  
12 ~~at its 01/28/1998 meeting as a PWG standard. Twelfth-This thirteenth~~  
13 ~~and Final draft MIB that incorporates the agreements reached at the JMP~~  
14 ~~PWG Meeting, on 12/5/97 01/30/98 in H-Amaui. on issues in V0.9087 which~~  
15 ~~was released after the 10/31-12/05/97 meeting. The changes include:~~

- 16 ~~1. Adding a mibs(1) arc between pwg(2699) and jobmonMIB(1)~~
- 17
- 18 ~~2. Added three job submission ID formats for IDPS.~~
- 19
- 20 ~~3. Minor typos.~~
- 21
- 22 ~~4. Add white space for certain MIB compilers.~~
- 23

24  
25  
26  
27 ~~This MIB is now an approved PWG standard and is being forwarded to the~~  
28 ~~Internet-Drafts DL as an Internet Draft. Then the JMP chair will~~  
29 ~~request that it be circulated as an Informational RFC which will~~  
30 ~~initiate a four week review by the IESG for submission as an~~  
31 ~~informational RFC.~~

- 32 ~~1.use the new PWG OIDs without the standard arc.~~
- 33
- 34 ~~2.make the document a PWG draft standard that will be sent as an~~  
35 ~~Internet Draft that will become an IETF Informational RFC,~~  
36 ~~including changing the IANA Considerations section~~
- 37
- 38 ~~3.add natural language support like IPP~~
- 39
- 40 ~~4.fix the issues with monitoring collated/uncollated~~  
41 ~~implementations~~
- 42
- 43 ~~5.fix impressions completed,~~
- 44
- 45 ~~6.allows multiple Job Submission Id entries to point to the same~~  
~~jmJobIndex entry~~
- 46
- 47 ~~7.and add 3 new Job Submission Ids~~
- 48
- 49 ~~8.Shortened processingMessageNaturalLanguageTag(7) to~~  
50 ~~processingMessageNaturalLangTag(7) so 31 characters.~~
- 51
- 52 ~~See the change history in the separate file: changes.doc .pdf.~~

46 ~~We agreed that the MIB specification is finished except for any~~  
47 ~~editorial comments that people may have. See the separate issues.doc~~  
48 ~~and .pdf file.~~

49 ~~I've also produced a variation on this document which has all variable~~  
50 ~~font (jmp mib.doc .pdf) without revision marks. This is the version~~  
51 ~~that the JMP should use to make comments. It has line numbers.~~

52 The MIB has been greatly simplified so that now there are only 18  
53 objects in the MIB. There are 73 attributes.

54  
55 INTERNET-DRAFT  
56  
57 R. Bergman  
58 Dataproducts Corp.  
59 T. Hastings  
60 Xerox Corporation  
61 S. Isaacson  
62 Novell, Inc.  
63 H. Lewis  
64 IBM Corp.  
65 ~~January 13~~February 3, 1998

64 Job Monitoring MIB - V1  
65 <draft-ietf-printmib-job-monitor-07.txt>

66  
67 Status of this Memo

68 This document is an Internet-Draft. Internet-Drafts are working  
69 documents of the Internet Engineering Task Force (IETF), its  
70 areas, and its working groups. Note that other groups may also  
71 distribute working documents as Internet-Drafts.

72 Internet-Drafts are draft documents valid for a maximum of six  
73 months and may be updated, replaced, or obsoleted by other  
74 documents at any time. It is inappropriate to use Internet-Drafts  
75 as reference material or to cite them other than as "work in  
76 progress."

77 To learn the current status of any Internet-Draft, please check  
78 the "lid-abstracts.txt" listing contained in the Internet-Drafts  
79 Shadow Directories on ftp.is.co.za (Africa), nic.nordu.net  
80 (Europe), munnari.oz.au (Pacific Rim), ds.internic.net (US East  
81 Coast), or ftp.isi.edu (US West Coast).

82 This Internet-Draft expires on ~~July 13~~August 3, 1998.

83  
84 Abstract

85 This document has been developed and approved by the Printer  
86 Working Group (PWG) as a PWG standard. It is intended to be  
87 distributed as an Informational RFC. This document provides a  
88 printer industry standard SNMP MIB for (1) monitoring the status  
89 and progress of print jobs (2) obtaining resource requirements  
90 before a job is processed, (3) monitoring resource consumption  
91 while a job is being processed and (4) collecting resource  
92 accounting data after the completion of a job. This MIB is  
93 intended to be implemented (1) in a printer or (2) in a server  
94 that supports one or more printers. Use of the object set is not  
95 limited to printing. However, support for services other than  
96 printing is outside the scope of this Job Monitoring MIB. Future  
97 extensions to this MIB may include, but are not limited to, fax  
98 machines and scanners.



99		
100		TABLE OF CONTENTS
101	1.	INTRODUCTION 10
102	1.1	Types of Information in the MIB 10
103	1.2	Types of Job Monitoring Applications 12
104	2.	TERMINOLOGY AND JOB MODEL 13
105	2.1	System Configurations for the Job Monitoring MIB 16
106	2.1.1	Configuration 1 - client-printer 16
107	2.1.2	Configuration 2 - client-server-printer - agent in the
108		server 17
109	2.1.3	Configuration 3 - client-server-printer - client monitors
110		printer agent and server 18
111	3.	MANAGED OBJECT USAGE 20
112	3.1	Conformance Considerations 20
113	3.1.1	Conformance Terminology 20
114	3.1.2	Agent Conformance Requirements 20
115	3.1.2.1	MIB II System Group objects 21
116	3.1.2.2	MIB II Interface Group objects 21
117	3.1.2.3	Printer MIB objects 21
118	3.1.3	Job Monitoring Application Conformance Requirements 21
119	3.2	The Job Tables and the Oldest Active and Newest Active Indexes 21
120	3.3	The Attribute Mechanism 23
121	3.3.1	Conformance of Attribute Implementation 24
122	3.3.2	Useful, 'Unknown', and 'Other' Values for Objects and
123		Attributes 24
124	3.3.3	Data Sub-types and Attribute Naming Conventions 25
125	3.3.4	Single-Value (Row) Versus Multi-Value (MULTI-ROW)
126		Attributes 26
127	3.3.5	Requested Objects and Attributes 26
128	3.3.6	Consumption Attributes 27
129	3.3.7	Index Value Attributes 27
130	3.4	Monitoring Job Progress 27
131	3.5	Job Identification 31
132	3.6	Internationalization Considerations 32
133	3.6.1	Text generated by the server or device 32
134	3.6.2	Text supplied by the job submitter 33
135	3.6.3	'DateAndTime' for representing the date and time 34

136	3.7 IANA and PWG Registration Considerations	34
137	3.7.1 PWG Registration of enums	35
138	3.7.1.1 Type 1 enumerations	35
139	3.7.1.2 Type 2 enumerations	35
140	3.7.1.3 Type 3 enumeration	36
141	3.7.2 PWG Registration of type 2 bit values	36
142	3.7.3 PWG Registration of Job Submission Id Formats	36
143	3.7.4 PWG Registration of MIME types/sub-types for document-	
144	formats	36
145	3.8 Security Considerations	36
146	3.8.1 Read-Write objects	36
147	3.8.2 Read-Only Objects In Other User's Jobs	37
148	3.9 Notifications	37
149	4. MIB SPECIFICATION	37
150	Textual Conventions for this MIB Module	37
151	JmUTF8StringTC	39
152	JmJobStringTC	39
153	JmNaturalLanguageTagTC	39
154	JmTimeStampTC	39
155	JmJobSourcePlatformTypeTC	40
156	JmFinishingTC	41
157	JmPrintQualityTC	42
158	JmPrinterResolutionTC	42
159	JmTonerEconomyTC	43
160	JmBooleanTC	43
161	JmMediumTypeTC	43
162	JmJobCollationTypeTC	45
163	JmJobSubmissionIDTypeTC	46
164	JmJobStateTC	50
165	JmAttributeTypeTC	53
166	other (Int32(-2..) and/or Octets63)	53
167	Job State attributes	54
168	jobStateReasons2 (JmJobStateReasons2TC)	54
169	jobStateReasons3 (JmJobStateReasons3TC)	54
170	jobStateReasons4 (JmJobStateReasons4TC)	54
171	processingMessage (UTF8String63)	54
172	processingMessageNaturalLangTag (Octets63)	55
173	jobCodedCharSet (CodedCharSet)	55
174	jobNaturalLanguageTag (Octets63)	56
175	Job Identification attributes	56
176	jobURI (Octets(0..63))	56
177	jobAccountName (Octets63)	56
178	serverAssignedJobName (JobString63)	57
179	jobName (JobString63)	57
180	jobServiceTypes (JmJobServiceTypesTC)	58
181	jobSourceChannelIndex (Int32(0..))	58
182	jobSourcePlatformType (JmJobSourcePlatformTypeTC)	58

183	submittingServerName (JobString63)	58
184	submittingApplicationName (JobString63)	58
185	jobOriginatingHost (JobString63)	59
186	deviceNameRequested (JobString63)	59
187	queueNameRequested (JobString63)	59
188	physicalDevice (hrDeviceIndex and/or UTF8String63)	59
189	numberOfDocuments (Int32(-2..))	59
190	fileName (JobString63)	60
191	documentName (JobString63)	60
192	jobComment (JobString63)	60
193	documentFormatIndex (Int32(0..))	60
194	documentFormat (PrtInterpreterLangFamilyTC and/or Octets63)	61
195	Job Parameter attributes	61
196	jobPriority (Int32(-2..100))	61
197	jobProcessAfterDateAndTime (DateAndTime)	62
198	jobHold (JmBooleanTC)	62
199	jobHoldUntil (JobString63)	62
200	outputBin (Int32(0..) and/or JobString63)	62
201	sides (Int32(-2..2))	63
202	finishing (JmFinishingTC)	63
203	Image Quality attributes (requested and used)	63
204	printQualityRequested (JmPrintQualityTC)	63
205	printQualityUsed (JmPrintQualityTC)	63
206	printerResolutionRequested (JmPrinterResolutionTC)	63
207	printerResolutionUsed (JmPrinterResolutionTC)	63
208	tonerEcomonyRequested (JmTonerEconomyTC)	63
209	tonerEcomonyUsed (JmTonerEconomyTC)	63
210	tonerDensityRequested (Int32(-2..100))	63
211	tonerDensityUsed (Int32(-2..100))	64
212	Job Progress attributes (requested and consumed)	64
213	jobCopiesRequested (Int32(-2..))	64
214	jobCopiesCompleted (Int32(-2..))	64
215	documentCopiesRequested (Int32(-2..))	64
216	documentCopiesCompleted (Int32(-2..))	64
217	jobKOctetsTransferred (Int32(-2..))	65
218	sheetCompletedCopyNumber (Int32(-2..))4	65
219	sheetCompletedDocumentNumber (Int32(-2..))4	65
220	jobCollationType JmJobCollationTypeTC)	65
221	Impression attributes (requested and consumed)	66
222	impressionsSpooled (Int32(-2..))	66
223	impressionsSentToDevice (Int32(-2..))	66
224	impressionsInterpreted (Int32(-2..))	66
225	impressionsCompletedCurrentCopy (Int32(-2..))	66
226	fullColorImpressionsCompleted (Int32(-2..))	66
227	highlightColorImpressionsCompleted (Int32(-2..))	67
228	Page attributes (requested and consumed)	67
229	pagesRequested (Int32(-2..))	67
230	pagesCompleted (Int32(-2..))	67
231	pagesCompletedCurrentCopy (Int32(-2..))	68
232	Sheet attributes (requested and consumed)	68
233	sheetsRequested (Int32(-2..))	68
234	sheetsCompleted (Int32(-2..))	68

235	sheetsCompletedCurrentCopy    (Int32(-2..))	68
236	Resource attributes (requested and consumed)	68
237	mediumRequested    (JmMediumTypeTC and/or JobString63)	69
238	mediumConsumed    (Int32(-2..) and/or JobString63)	69
239	colorantRequested    (Int32(-2..) and/or JobString63)	69
240	colorantConsumed    (Int32(-2..) and/or JobString63)	70
241	Time attributes (set by server or device)	70
242	jobSubmissionToServerTime    (JmTimeStampTC and/or DateAndTime)	70
243	jobSubmissionTime    (JmTimeStampTC and/or DateAndTime)	70
244	jobStartedBeingHeldTime    (JmTimeStampTC)	71
245	jobStartedProcessingTime    (JmTimeStampTC and/or DateAndTime)	71
246	jobCompletionTime    (JmTimeStampTC and/or DateAndTime)	71
247	jobProcessingCPUTime    (Int32(-2..))	71
248	JmJobServiceTypesTC	73
249	JmJobStateReasons1TC	75
250	JmJobStateReasons2TC	79
251	JmJobStateReasons3TC	83
252	JmJobStateReasons4TC	83
253	The General Group (MANDATORY)	84
254	jmGeneralJobSetIndex    (Int32(1..32767))	85
255	jmGeneralNumberOfActiveJobs    (Int32(0..))	85
256	jmGeneralOldestActiveJobIndex    (Int32(0..))	86
257	jmGeneralNewestActiveJobIndex    (Int32(0..))	86
258	jmGeneralJobPersistence    (Int32(15..))	87
259	jmGeneralAttributePersistence    (Int32(15..))	87
260	jmGeneralJobSetName    (UTF8String63)	88
261	The Job ID Group (MANDATORY)	88
262	jmJobSubmissionID    (OCTET STRING(SIZE(48)))	90
263	jmJobIDJobSetIndex    (Int32(0..32767))	91
264	jmJobIDJobIndex    (Int32(0..))	91
265	The Job Group (MANDATORY)	91
266	jmJobIndex    (Int32(1..))	93
267	jmJobState    (JmJobStateTC)	93
268	jmJobStateReasons1    (JmJobStateReasons1TC)	94
269	jmNumberOfInterveningJobs    (Int32(-2..))	94
270	jmJobKOctetsPerCopyRequested    (Int32(-2..))	95
271	jmJobKOctetsProcessed    (Int32(-2..))	95
272	jmJobImpressionsPerCopyRequested    (Int32(-2..))	96
273	jmJobImpressionsCompleted    (Int32(-2..))	96
274	jmJobOwner    (JobString63)	97
275	The Attribute Group (MANDATORY)	97
276	jmAttributeTypeIndex    (JmAttributeTypeTC)	100
277	jmAttributeInstanceIndex    (Int32(1..32767))	100
278	jmAttributeValueAsInteger    (Int32(-2..))	101
279	jmAttributeValueAsOctets    (Octets63)	102
280	5. APPENDIX A - IMPLEMENTING THE JOB LIFE CYCLE	105



281	6.	APPENDIX B - SUPPORT OF JOB SUBMISSION PROTOCOLS	106
282	7.	REFERENCES	106
283	8.	AUTHOR'S ADDRESSES	108
284	9.	INDEX	111
285			

## 286 Job Monitoring MIB

## 287 1. Introduction

288 This specification defines an official Printer Working Group (PWG)  
289 [PWG] standard SNMP MIB for the monitoring of jobs on network printers.  
290 This specification is being published as an IETF Information Document  
291 for the convenience of the Internet community. In consultation with  
292 the IETF Application Area Directors, it was concluded that this MIB  
293 specification properly belongs as an Information document, because this  
294 MIB monitors a service node on the network, rather than a network node  
295 proper.

296 The Job Monitoring MIB is intended to be implemented by an agent within  
297 a printer or the first server closest to the printer, where the printer  
298 is either directly connected to the server only or the printer does not  
299 contain the job monitoring MIB agent. It is recommended that  
300 implementations place the SNMP agent as close as possible to the  
301 processing of the print job. This MIB applies to printers with and  
302 without spooling capabilities. This MIB is designed to be compatible  
303 with most current commonly-used job submission protocols. In most  
304 environments that support high function job submission/job control  
305 protocols, like ISO DPA[iso-dpa], those protocols would be used to  
306 monitor and manage print jobs rather than using the Job Monitoring MIB.

307 The Job Monitoring MIB consists of a General Group, a Job Submission ID  
308 Group, a Job Group, and an Attribute Group. Each group is a table.  
309 All accessible objects are read-only. The General Group contains  
310 general information that applies to all jobs in a job set. The Job  
311 Submission ID table maps the job submission ID that the client uses to  
312 identify a job to the jmJobIndex that the Job Monitoring Agent uses to  
313 identify jobs in the Job and Attribute tables. The Job table contains  
314 the MANDATORY integer job state and status objects. The Attribute  
315 table consists of multiple entries per job that specify (1) job and  
316 document identification and parameters, (2) requested resources, and  
317 (3) consumed resources during and after job processing/printing. A  
318 larger number of job attributes are defined as textual conventions that  
319 an agent SHALL return if the server or device implements the  
320 functionality so represented and the agent has access to the  
321 information.

## 322 1.1 Types of Information in the MIB

323 The job MIB is intended to provide the following information for the  
324 indicated Role Models in the Printer MIB[print-mib] (Appendix D - Roles  
325 of Users).

326 User:

327 Provide the ability to identify the least busy printer. The user  
328 will be able to determine the number and size of jobs waiting for  
329 each printer. No attempt is made to actually predict the length  
330 of time that jobs will take.

331 Provide the ability to identify the current status of the user's  
332 job (user queries).

333 Provide a timely indication that the job has completed and where  
334 it can be found.

335 Provide error and diagnostic information for jobs that did not  
336 successfully complete.

337 Operator:

338 Provide a presentation of the state of all the jobs in the print  
339 system.

340 Provide the ability to identify the user that submitted the print  
341 job.

342 Provide the ability to identify the resources required by each  
343 job.

344 Provide the ability to define which physical printers are  
345 candidates for the print job.

346 Provide some idea of how long each job will take. However, exact  
347 estimates of time to process a job is not being attempted.  
348 Instead, objects are included that allow the operator to be able  
349 to make gross estimates.

350 Capacity Planner:

351 Provide the ability to determine printer utilization as a  
352 function of time.

353 Provide the ability to determine how long jobs wait before  
354 starting to print.

355 Accountant:

356 Provide information to allow the creation of a record of  
357 resources consumed and printer usage data for charging users or  
358 groups for resources consumed.

359 Provide information to allow the prediction of consumable usage  
360 and resource need.

361 The MIB supports printers that can contain more than one job at a time,  
362 but still be usable for low end printers that only contain a single job  
363 at a time. In particular, the MIB supports the needs of Windows and  
364 other PC environments for managing low-end direct-connect (serial or  
365 parallel) and networked devices without unnecessary overhead or  
366 complexity, while also providing for higher end systems and devices.

## 367 1.2 Types of Job Monitoring Applications

368 The Job Monitoring MIB is designed for the following types of  
369 monitoring applications:

- 370 1. Monitor a single job starting when the job is submitted and  
371 ending a defined period after the job completes. The Job  
372 Submission ID table provides the map to find the specific job  
373 to be monitored.
- 374 2. Monitor all 'active' jobs in a queue, which this specification  
375 generalizes to a "job set". End users may use such a program  
376 when selecting a least busy printer, so the MIB is designed for  
377 such a program to start up quickly and find the information  
378 needed quickly without having to read all (completed) jobs in  
379 order to find the active jobs. System operators may also use  
380 such a program, in which case it would be running for a long  
381 period of time and may also be interested in the jobs that have  
382 completed. Finally such a program may be used to provide an  
383 enhanced console and logging capability.
- 384 3. Collect resource usage for accounting or system utilization  
385 purposes that copy the completed job statistics to an  
386 accounting system. It is recognized that depending on  
387 accounting programs to copy MIB data during the job-retention  
388 period is somewhat unreliable, since the accounting program may  
389 not be running (or may have crashed). Such a program is also  
390 expected to keep a shadow copy of the entire Job Attribute  
391 table including completed, canceled, and aborted jobs which the  
392 program updates on each polling cycle. Such a program polls at  
393 the rate of the persistence of the Attribute table. The design  
394 is not optimized to help such an application determine which  
395 jobs are completed, canceled, or aborted. Instead, the  
396 application SHALL query each job that the application's shadow  
397 copy shows was not complete, canceled, or aborted at the  
398 previous poll cycle to see if it is now complete or canceled,  
399 plus any new jobs that have been submitted.

400 The MIB provides a set of objects that represent a compatible subset of  
401 job and document attributes of the ISO DPA standard[iso-dpa] and the  
402 Internet Printing Protocol (IPP)[ipp-model], so that coherence is  
403 maintained between these two protocols and the information presented to  
404 end users and system operators by monitoring applications. However,  
405 the job monitoring MIB is intended to be used with printers that  
406 implement other job submitting and management protocols, such as IEEE  
407 1284.1 (TIPSI)[tipsi], as well as with ones that do implement ISO DPA.

408 Thus the job monitoring MIB does not require implementation of either  
409 the ISO DPA or IPP protocols.

410 The MIB is designed so that an additional MIB(s) can be specified in  
411 the future for monitoring multi-function (scan, FAX, copy) jobs as an  
412 augmentation to this MIB.

## 413 2. Terminology and Job Model

414 This section defines the terms that are used in this specification and  
415 the general model for jobs in alphabetical order.

416 NOTE - Existing systems use conflicting terms, so these terms are  
417 drawn from the ISO 10175 Document Printing Application (DPA)  
418 standard[iso-dpa]. For example, PostScript systems use the term  
419 *session* for what is called a *job* in this specification and the term  
420 *job* to mean what is called a *document* in this specification.

421 Accounting Application: The SNMP management application that copies  
422 job information to some more permanent medium so that another  
423 application can perform accounting on the data for Accountants, Asset  
424 Managers, and Capacity Planners use.

425 Agent: The network entity that accepts SNMP requests from a *monitor* or  
426 *accounting application* and provides access to the instrumentation for  
427 managing jobs modeled by the management objects defined in the Job  
428 Monitoring MIB module for a *server* or a *device*.

429 Attribute: A name, value-pair that specifies a job or document  
430 instruction, a status, or a condition of a job or a document that has  
431 been submitted to a server or device. A particular attribute NEED NOT  
432 be present in each job instance. In other words, attributes are  
433 present in a job instance only when there is a need to express the  
434 value, either because (1) the client supplied a value in the job  
435 submission protocol, (2) the document data contained an embedded  
436 attribute, or (3) the server or device supplied a default value. An  
437 agent SHALL represent an attribute as an entry (row) in the Attribute  
438 table in this MIB in which entries are present only when necessary.  
439 Attributes are identified in this MIB by an enum.

440 Client: The network entity that *end users* use to submit jobs to  
441 *spoolers, servers, or printers* and other *devices*, depending on the  
442 configuration, using any job submission protocol over a serial or  
443 parallel port to a directly-connected device or over the network to a  
444 networked-connected device.

445 Device: A hardware entity that (1) interfaces to humans, such as a  
446 device that produces marks on paper or scans marks on paper to produce  
447 an electronic representation, (2) accesses digital media, such as CD-  
448 ROMs, or (3) interfaces electronically to another device, such as sends  
449 FAX data to another FAX device.

450 Document: A sub-section within a job that contains print data and  
451 *document instructions* that apply to just the document.

452 Document Instruction: An instruction specifying how to process the  
453 document. Document instructions MAY be passed in the job submission  
454 protocol separate from the actual document data, or MAY be embedded in  
455 the document data or a combination, depending on the job submission  
456 protocol and implementation.

457 End User: A user that uses a client to submit a print job. See  
458 "user".

459 Impression: For a print job, an impression is the passage of the  
460 entire side of a sheet by the marker, whether or not any marks are made  
461 and independent of the number of passes that the side makes past the  
462 marker. Thus a four pass color process counts as a single impression,  
463 as does highlight color. Impression counters count all kinds:  
464 monochrome, highlight color, and full process color, while full color  
465 counters only count full color impressions, and high light color  
466 counters only count high light color impressions.

467 One-sided processing involves one impression per sheet. Two-sided  
468 processing involves two impressions per sheet. If a two-sided document  
469 has an odd number of pages, the last sheet still counts as two  
470 impressions, if that sheet makes two passes through the marker or the  
471 marker marks on both sides of a sheet in a single pass. Two-up  
472 printing is the placement of two logical pages on one side of a sheet  
473 and so is still a single impression. See "page" and "sheet".

474 NOTE - Since impressions include blank sides, it is suggested that  
475 accounting application implementers consider charging for sheets,  
476 rather than impressions, possibly using the value of the sides  
477 attribute to select different charges for one-sided versus two-sided  
478 printing, since some users may think that impressions don't include  
479 blank sides.

480 Internal Collation: The production of the sheets for each document copy  
481 performed within the printing device by making multiple passes over  
482 either the source or an intermediate representation of the document.

483 Job: A unit of work whose results are expected together without  
484 interjection of unrelated results. A job contains one or more  
485 *documents*.

486 Job Accounting: The activity of a management application of accessing  
487 the MIB and recording what happens to the job during and after the  
488 processing of the job.

489 Job Instruction: An instruction specifying how, when, or where the job  
490 is to be processed. Job instructions MAY be passed in the job  
491 submission protocol or MAY be embedded in the document data or a  
492 combination depending on the job submission protocol and  
493 implementation.

494 Job Monitoring (using SNMP): The activity of a management application  
495 of accessing the MIB and (1) identifying jobs in the job tables being  
496 processed by the server, printer or other devices, and (2) displaying  
497 information to the user about the processing of the job.

498 Job Monitoring Application: The SNMP management application that End  
499 Users, and System Operators use to monitor jobs using SNMP. A monitor  
500 MAY be either a separate application or MAY be part of the client that  
501 also submits jobs. See "monitor".

502 Job Set: A group of jobs that are queued and scheduled together  
503 according to a specified scheduling algorithm for a specified device or  
504 set of devices. For implementations that embed the SNMP agent in the  
505 device, the MIB job set normally represents *all* the jobs known to the  
506 device, so that the implementation only implements a single job set.  
507 If the SNMP agent is implemented in a server that controls one or more  
508 devices, each MIB job set represents a job queue for (1) a specific  
509 device or (2) set of devices, if the server uses a single queue to load  
510 balance between several devices. Each job set is disjoint; no job  
511 SHALL be represented in more than one MIB job set.

512 Monitor: Short for Job Monitoring Application.

513 Page: A page is a logical division of the original source document.  
514 Number up is the imposition of more than one page on a single side of a  
515 sheet. See "impression" and "sheet" and "two-up".

516 Proxy: An agent that acts as a concentrator for one or more other  
517 agents by accepting SNMP operations on the behalf of one or more other  
518 agents, forwarding them on to those other agents, gathering responses  
519 from those other agents and returning them to the original requesting  
520 monitor.

521 Queuing: The act of a *device* or *server* of ordering (queuing) the jobs  
522 for the purposes of scheduling the jobs to be processed.

523 Printer: A *device* that puts marks on media.

524 Server: A network entity that accepts jobs from clients and in turn  
525 submits the jobs to *printers* and other *devices* that may be directly  
526 connected to the server via a serial or parallel port or may be on the  
527 network. A server MAY be a printer *supervisor* control program, or a  
528 print *spooler*.

529 Sheet: A sheet is a single instance of a medium, whether printing on  
530 one or both sides of the medium. See "impression" and "page".



531 SNMP Information Object: A name, value-pair that specifies an action,  
532 a status, or a condition in an SNMP MIB. Objects are identified in  
533 SNMP by an OBJECT IDENTIFIER.

534 Spooler: A server that accepts jobs, spools the data, and decides when  
535 and on which printer to print the job. A spooler is a client to a  
536 printer or a printer supervisor, depending on implementation.

537 Spooling: The act of a *device* or *server* of (1) accepting jobs and (2)  
538 writing the job's attributes and document data on to secondary storage.

539 Stacked: When a media sheet is placed in an output bin of a device.

540 Supervisor: A server that contains a control program that controls a  
541 printer or other device. A supervisor is a client to the printer or  
542 other device.

543 System Operator: A user that uses a monitor to monitor the system and  
544 carries out tasks to keep the system running.

545 System Administrator: A user that specifies policy for the system.

546 Two-up: The placement of two pages on one side of a sheet so that each  
547 side or impressions counts as two pages. See "page" and "sheet".

548 User: A person that uses a client or a monitor. See "end user".

549 2.1 System Configurations for the Job Monitoring MIB

550 This section enumerates the three configurations in which the Job  
551 Monitoring MIB is intended to be used. To simplify the pictures, the  
552 *devices* are shown as *printers*. See section 1.1 entitled "Types of  
553 Information in the MIB".

554 The diagram in the Printer MIB[print-mib] entitled: "One Printer's View  
555 of the Network" is assumed for this MIB as well. Please refer to that  
556 diagram to aid in understanding the following system configurations.

557 2.1.1 Configuration 1 - client-printer

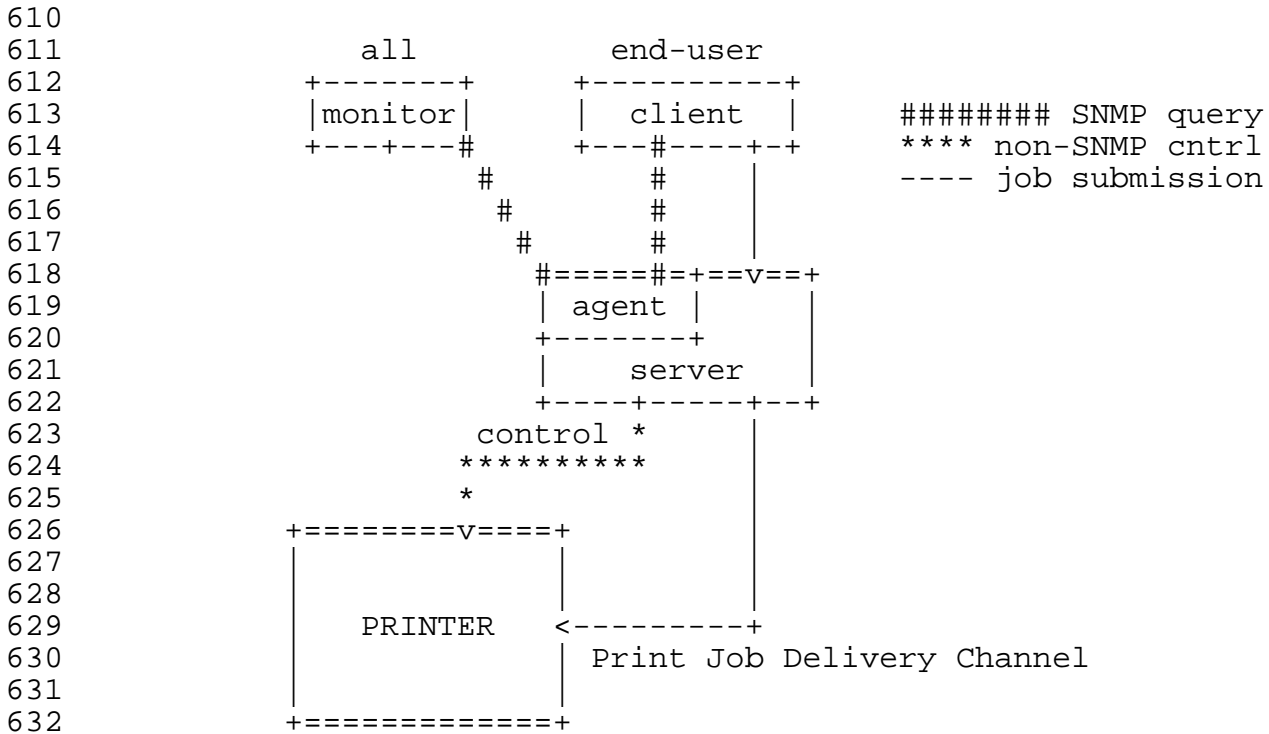
558 In the client-printer configuration 1, the client(s) submit jobs  
559 directly to the printer, either by some direct connect, or by network  
560 connection.

561 The job submitting client and/or monitoring application monitor jobs by  
562 communicating directly with an agent that is part of the printer. The  
563 agent in the printer SHALL keep the job in the Job Monitoring MIB as  
564 long as the job is in the printer, plus a defined time period after the  
565 job enters the completed state in which accounting programs can copy  
566 out the accounting data from the Job Monitoring MIB.

567







633 Figure 2-2 - Configuration 2 - client-server-printer - agent in the  
 634 server

635 The Job Monitoring MIB is designed to support the following  
 636 relationships (not shown in Figure 2-2):

- 637 1. Multiple clients MAY submit jobs to a server.
- 638 2. Multiple clients MAY monitor a server.
- 639 3. Multiple monitors MAY monitor a server.
- 640 4. A client MAY submit jobs to multiple servers.
- 641 5. A monitor MAY monitor multiple servers.
- 642 6. Multiple servers MAY submit jobs to a printer.
- 643 7. Multiple servers MAY control a printer.

644 2.1.3 Configuration 3 - client-server-printer - client monitors printer  
 645 agent and server

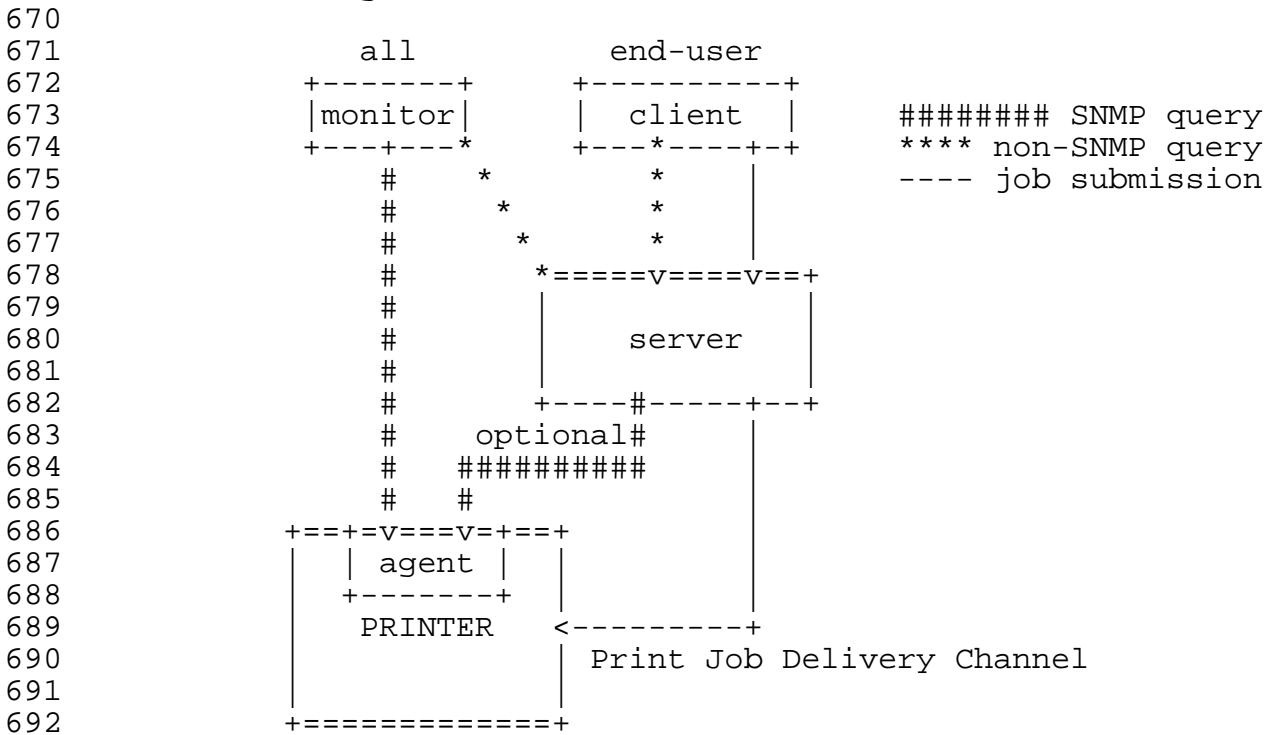
646 In the client-server-printer configuration 3, the client(s) submit jobs  
 647 to an intermediate server by some network connection, *not* directly to  
 648 the printer. That server does *not* contain a Job Monitoring MIB agent.

649 The job submitting client and/or monitoring application monitor jobs by  
 650 communicating directly with:

- 651 1. The server using some undefined protocol to monitor jobs in the  
 652 server (that does not contain the Job Monitoring MIB) AND
- 653 2. A Job Monitoring MIB agent that is part of the printer to  
 654 monitor jobs after the server passes the jobs to the printer.  
 655 In such configurations, the server deletes its copy of the job

656 from the server after submitting the job to the printer usually  
 657 almost immediately (before the job does much processing, if  
 658 any).

659 In configuration 3, the agent (in the printer) SHALL keep the values of  
 660 the objects in the Job Monitoring MIB that the agent implements updated  
 661 for a job that the server has submitted to the printer. The agent  
 662 SHALL obtain information about the jobs submitted to the printer from  
 663 the server (either in the job submission protocol, in the document  
 664 data, or by direct query of the server), in order to populate some of  
 665 the objects the Job Monitoring MIB in the printer. The agent in the  
 666 printer SHALL keep the job in the Job Monitoring MIB as long as the job  
 667 is in the Printer, and longer in order to implement the completed state  
 668 in which monitoring programs can copy out the accounting data from the  
 669 Job Monitoring MIB.



693 Figure 2-3 - Configuration 3 - client-server-printer - client monitors  
 694 printer agent and server

695 The Job Monitoring MIB is designed to support the following  
 696 relationships (not shown in Figure 2-3):

- 697 1. Multiple clients MAY submit jobs to a server.
- 698 2. Multiple clients MAY monitor a server.
- 699 3. Multiple monitors MAY monitor a server.
- 700 4. A client MAY submit jobs to multiple servers.
- 701 5. A monitor MAY monitor multiple servers.
- 702 6. Multiple servers MAY submit jobs to a printer.
- 703 7. Multiple servers MAY control a printer.

## 704 3. Managed Object Usage

705 This section describes the usage of the objects in the MIB.

## 706 3.1 Conformance Considerations

707 In order to achieve interoperability between job monitoring  
708 applications and job monitoring agents, this specification includes the  
709 conformance requirements for both monitoring applications and agents.

## 710 3.1.1 Conformance Terminology

711 This specification uses the verbs: "SHALL", "SHOULD", "MAY", and "NEED  
712 NOT" to specify conformance requirements according to RFC 2119 [req-  
713 words] as follows:

714 "SHALL": indicates an action that the subject of the sentence must  
715 implement in order to claim conformance to this specification

716 "MAY": indicates an action that the subject of the sentence does not  
717 have to implement in order to claim conformance to this  
718 specification, in other words that action is an implementation option

719 "NEED NOT": indicates an action that the subject of the sentence  
720 does not have to implement in order to claim conformance to this  
721 specification. The verb "NEED NOT" is used instead of "may not",  
722 since "may not" sounds like a prohibition.

723 "SHOULD": indicates an action that is recommended for the subject of  
724 the sentence to implement, but is not required, in order to claim  
725 conformance to this specification.

## 726 3.1.2 Agent Conformance Requirements

727 A conforming agent:

- 728 1. SHALL implement *all* MANDATORY groups in this specification.
- 729 2. SHALL implement any attributes if (1) the server or device  
730 supports the functionality represented by the attribute and (2)  
731 the information is available to the agent.
- 732 3. SHOULD implement both forms of an attribute if it implements an  
733 attribute that permits a choice of INTEGER and OCTET STRING  
734 forms, since implementing both forms may help management  
735 applications by giving them a choice of representations, since  
736 the representation are equivalent. See the JmAttributeTypeTC  
737 textual-convention.

738 NOTE - This MIB, like the Printer MIB, is written following the subset  
739 of SMIV2 that can be supported by SMIV1 and SNMPV1 implementations.

## 740 3.1.2.1 MIB II System Group objects

741 The Job Monitoring MIB agent SHALL implement all objects in the System  
742 Group of MIB-II[mib-II], whether the Printer MIB[print-mib] is  
743 implemented or not.

## 744 3.1.2.2 MIB II Interface Group objects

745 The Job Monitoring MIB agent SHALL implement all objects in the  
746 Interfaces Group of MIB-II[mib-II], whether the Printer MIB[print-mib]  
747 is implemented or not.

## 748 3.1.2.3 Printer MIB objects

749 If the agent is providing access to a device that is a printer, the  
750 agent SHALL implement all of the MANDATORY objects in the Printer  
751 MIB[print-mib] and all the objects in other MIBs that conformance to  
752 the Printer MIB requires, such as the Host Resources MIB[hr-mib]. If  
753 the agent is providing access to a server that controls one or more  
754 direct-connect or networked printers, the agent NEED NOT implement the  
755 Printer MIB and NEED NOT implement the Host Resources MIB.

## 756 3.1.3 Job Monitoring Application Conformance Requirements

757 A conforming job monitoring application:

- 758 1. SHALL accept the full syntactic range for all objects in all  
759 MANDATORY groups and all MANDATORY attributes that are required  
760 to be implemented by an agent according to Section 3.1.2 and  
761 SHALL either present them to the user or ignore them.
- 762 2. SHALL accept the full syntactic range for *all* attributes,  
763 including enum and bit values specified in this specification  
764 and additional ones that may be registered with the PWG and  
765 SHALL either present them to the user or ignore them. In  
766 particular, a conforming job monitoring application SHALL not  
767 malfunction when receiving any standard or registered enum or  
768 bit values. See Section 3.7 entitled "IANA and PWG  
769 Registration Considerations".
- 770 3. SHALL NOT fail when operating with agents that materialize  
771 attributes *after* the job has been submitted, as opposed to when  
772 the job is submitted.
- 773 4. SHALL, if it supports a time attribute, accept either form of  
774 the time attribute, since agents are free to implement either  
775 time form.

## 776 3.2 The Job Tables and the Oldest Active and Newest Active Indexes

777 The jmJobTable and jmAttributeTable contain objects and attributes,  
778 respectively, for each job in a job set. These first two indexes are:

- 779 1. jmGeneralJobSetIndex - which job set
- 780 2. jmJobIndex - which job in the job set

781 In order for a monitoring application to quickly find that active jobs  
782 (jobs in the pending, processing, or processingStopped states), the MIB  
783 contains two indexes:

- 784 1. jmGeneralOldestActiveJobIndex - the index of the active job  
785 that has been in the tables the longest.
- 786 2. jmGeneralNewestActiveJobIndex - the index of the active job  
787 that has been most recently added to the tables.

788 The agent SHALL assign the next incremental value of jmJobIndex to the  
789 job, when a new job is accepted by the server or device to which the  
790 agent is providing access. If the incremented value of jmJobIndex  
791 would exceed the implementation-defined maximum value for jmJobIndex,  
792 the agent SHALL 'wrap' back to 1. An agent uses the resulting value of  
793 jmJobIndex for storing information in the jmJobTable and the  
794 jmAttributeTable about the job.

795 It is recommended that the largest value for jmJobIndex be much larger  
796 than the maximum number of jobs that the implementation can contain at  
797 a single time, so as to minimize the premature re-use of a jmJobIndex  
798 value for a newer job while clients retain the same 'stale' value for  
799 an older job.

800 It is recommended that agents that are providing access to  
801 servers/devices that already allocate job-identifiers for jobs as  
802 integers use the same integer value for the jmJobIndex. Then  
803 management applications using this MIB and applications using other  
804 protocols will see the same job identifiers for the same jobs. Agents  
805 providing access to systems that contain jobs with a job identifier of  
806 0 SHALL map the job identifier value 0 to a jmJobIndex value that is  
807 one higher than the highest job identifier value that any job can have  
808 on that system. Then only job 0 will have a different job-identifier  
809 value than the job's jmJobIndex value.

810 NOTE - If a server or device accepts jobs using multiple job submission  
811 protocols, it may be difficult for the agent to meet the recommendation  
812 to use the job-identifier values that the server or device assigns as  
813 the jmJobIndex value, unless the server/device assigns job-identifiers  
814 for each of its job submission protocols from the same job-identifier  
815 number space.

816 Each time a new job is accepted by the server or device that the agent  
817 is providing access to AND that job is to be 'active' (pending,  
818 processing, or processingStopped, but not pendingHeld), the agent SHALL  
819 copy the value of the job's jmJobIndex to the  
820 jmGeneralNewestActiveJobIndex object. If the new job is to be  
821 'inactive' (pendingHeld state), the agent SHALL not change the value of  
822 jmGeneralNewestActiveJobIndex object (though the agent SHALL assign the  
823 next incremental jmJobIndex value to the job).

824 When a job transitions from one of the 'active' job states (pending,  
825 processing, processingStopped) to one of the 'inactive' job states  
826 (pendingHeld, completed, canceled, or aborted), with a jmJobIndex value  
827 that matches the jmGeneralOldestActiveJobIndex object, the agent SHALL  
828 advance (or wrap) the value to the next oldest 'active' job, if any.  
829 See the JmJobStateTC textual-convention for a definition of the job  
830 states.

831 Whenever a job transitions from one of the 'inactive' job states to one  
832 of the 'active' job states (from pendingHeld to pending or processing),  
833 the agent SHALL update the value of either the  
834 jmGeneralOldestActiveJobIndex or the jmGeneralNewestActiveJobIndex  
835 objects, or both, if the job's jmJobIndex value is outside the range  
836 between jmGeneralOldestActiveJobIndex and  
837 jmGeneralNewestActiveJobIndex.

838 When all jobs become 'inactive', i.e., enter the pendingHeld,  
839 completed, canceled, or aborted states, the agent SHALL set the value  
840 of both the jmGeneralOldestActiveJobIndex and  
841 jmGeneralNewestActiveJobIndex objects to 0.

842 NOTE - Applications that wish to efficiently access all of the active  
843 jobs MAY use jmGeneralOldestActiveJobIndex value to start with the  
844 oldest active job and continue until they reach the index value equal  
845 to jmGeneralNewestActiveJobIndex, skipping over any pendingHeld,  
846 completed, canceled, or aborted jobs that might intervene.

847 If an application detects that the jmGeneralNewestActiveJobIndex is  
848 smaller than jmGeneralOldestActiveJobIndex, the job index has wrapped.  
849 In this case, the application SHALL reset the index to 1 when the end  
850 of the table is reached and continue the GetNext operations to find the  
851 rest of the active jobs.

852 NOTE - Applications detect the end of the jmAttributeTable table when  
853 the OID returned by the GetNext operation is an OID in a different MIB.  
854 There is no object in this MIB that specifies the maximum value for the  
855 jmJobIndex supported by the implementation.

856 When the server or device is power-cycled, the agent SHALL remember the  
857 next jmJobIndex value to be assigned, so that new jobs are not assigned  
858 the same jmJobIndex as recent jobs before the power cycle.

### 859 3.3 The Attribute Mechanism

860 Attributes are similar to information objects, except that attributes  
861 are identified by an enum, instead of an OID, so that attributes may be  
862 registered without requiring a new MIB. Also an implementation that  
863 does not have the functionality represented by the attribute can omit  
864 the attribute entirely, rather than having to return a distinguished  
865 value. The agent is free to materialize an attribute in the  
866 jmAttributeTable as soon as the agent is aware of the value of the  
867 attribute.



868 The agent materializes job attributes in a four-indexed  
869 jmAttributeTable:

- 870 1. jmGeneralJobSetIndex - which job set
- 871 2. jmJobIndex - which job in the job set
- 872 3. jmAttributeTypeIndex - which attribute
- 873 4. jmAttributeInstanceIndex - which attribute instance for those  
874 attributes that can have multiple values per job.

875 Some attributes represent information about a job, such as a file-name,  
876 a document-name, a submission-time or a completion time. Other  
877 attributes represent resources required, e.g., a medium or a colorant,  
878 etc. to process the job before the job starts processing OR to indicate  
879 the amount of the resource consumed during and after processing, e.g.,  
880 pages completed or impressions completed. If both a required and a  
881 consumed value of a resource is needed, this specification assigns two  
882 separate attribute enums in the textual convention.

883 NOTE - The table of contents lists all the attributes in order. This  
884 order is the order of enum assignments which is the order that the SNMP  
885 GetNext operation returns attributes. Most attributes apply to all  
886 three configurations covered by this MIB specification (see section 2.1  
887 entitled "System Configurations for the Job Monitoring MIB"). Those  
888 attributes that apply to a particular configuration are indicated as  
889 'Configuration n:' and SHALL NOT be used with other configurations.

### 890 3.3.1 Conformance of Attribute Implementation

891 An agent SHALL implement any attribute if (1) the server or device  
892 supports the functionality represented by the attribute and (2) the  
893 information is available to the agent. The agent MAY create the  
894 attribute row in the jmAttributeTable when the information is available  
895 or MAY create the row earlier with the designated 'unknown' value  
896 appropriate for that attribute. See next section.

897 If the server or device does not implement or does not provide access  
898 to the information about an attribute, the agent SHOULD NOT create the  
899 corresponding row in the jmAttributeTable.

### 900 3.3.2 Useful, 'Unknown', and 'Other' Values for Objects and Attributes

901 Some attributes have a 'useful' Integer32 value, some have a 'useful'  
902 OCTET STRING value, some MAY have either or both depending on  
903 implementation, and some MUST have both. See the JmAttributeTypeTC  
904 textual convention for the specification of each attribute.

905 SNMP requires that if an object cannot be implemented because its  
906 values cannot be accessed, then a compliant agent SHALL return an SNMP  
907 error in SNMPv1 or an exception value in SNMPv2. However, this MIB has  
908 been designed so that 'all' objects can and SHALL be implemented by an  
909 agent, so that neither the SNMPv1 error nor the SNMPv2 exception value



910 SHALL be generated by the agent. This MIB has also been designed so  
911 that when an agent materializes an attribute, the agent SHALL  
912 materialize a row consisting of both the jmAttributeValueAsInteger and  
913 jmAttributeValueAsOctets objects.

914 In general, values for objects and attributes have been chosen so that  
915 a management application will be able to determine whether a 'useful',  
916 'unknown', or 'other' value is available. When a useful value is not  
917 available for an object that agent SHALL return a zero-length string  
918 for octet strings, the value 'unknown(2)' for enums, a '0' value for an  
919 object that represents an index in another table, and a value '-2' for  
920 counting integers.

921 Since each attribute is represented by a row consisting of both the  
922 jmAttributeValueAsInteger and jmAttributeValueAsOctets MANDATORY  
923 objects, SNMP requires that the agent SHALL always create an attribute  
924 row with both objects specified. However, for most attributes the  
925 agent SHALL return a "useful" value for one of the objects and SHALL  
926 return the 'other' value for the other object. For integer only  
927 attributes, the agent SHALL always return a zero-length string value  
928 for the jmAttributeValueAsOctets object. For octet string only  
929 attributes, the agent SHALL always return a '-1' value for the  
930 jmAttributeValueAsInteger object.

### 931 3.3.3 Data Sub-types and Attribute Naming Conventions

932 Many attributes are sub-typed to give a more specific data type than  
933 Integer32 or OCTET STRING. The data sub-type of each attribute is  
934 indicated on the first line(s) of the description. Some attributes  
935 have several different data sub-type representations. When an  
936 attribute has both an Integer32 data sub-type and an OCTET STRING data  
937 sub-type, the attribute can be represented in a single row in the  
938 jmAttributeTable. In this case, the data sub-type name is not included  
939 as the last part of the name of the attribute, e.g., documentFormat(38)  
940 which is both an enum and/or a name. When the data sub-types cannot be  
941 represented by a single row in the jmAttributeTable, each such  
942 representation is considered a separate attribute and is assigned a  
943 separate name and enum value. For these attributes, the name of the  
944 data sub-type is the last part of the name of the attribute: Name,  
945 Index, DateAndTime, TimeStamp, etc. For example,  
946 documentFormatIndex(37) is an index.

947 NOTE: The Table of Contents also lists the data sub-type and/or data  
 948 sub-types of each attribute, using the textual-convention name when  
 949 such is defined. The following abbreviations are used in the Table of  
 950 Contents as shown:  
 951

'Int32(-2..)'	Integer32_(-2..2147483647)
'Int32(0..)'	Integer32_(0..2147483647)
'Int32(1..)'	Integer32_(1..2147483647)
'Int32(m..n)'	For all other Integer ranges, the lower and upper bound of the range is indicated.
'UTF8String63'	JmUTF8StringTC_(SIZE(0..63))
'JobString63'	JmJobStringTC_(SIZE(0..63))
'Octets63'	OCTET STRING_(SIZE(0..63))
'Octets(m..n)'	For all other OCTET STRING ranges, the exact range is indicated.

#### 952 3.3.4 Single-Value (Row) Versus Multi-Value (MULTI-ROW) Attributes

953 Most attributes SHALL have only one row per job. However, a few  
 954 attributes can have multiple values per job or even per document, where  
 955 each value is a separate row in the jmAttributeTable. Unless indicated  
 956 with 'MULTI-ROW:' in the JmAttributeTypeTC description, an agent SHALL  
 957 ensure that each attribute occurs only once in the jmAttributeTable for  
 958 a job. Most of the 'MULTI-ROW' attributes do not allow duplicate  
 959 values, i.e., the agent SHALL ensure that each value occurs only once  
 960 for a job. Only if the specification of the 'MULTI-ROW' attribute also  
 961 says "~~the values NEED NOT be unique~~There is no restriction on the same  
 962 xxx occurring in multiple rows" can the agent allow duplicate values to  
 963 occur for the job.

964 NOTE - Duplicates are allowed for 'extensive' 'MULTI-ROW' attributes,  
 965 such as fileName(34) or documentName(35) which are specified to be  
 966 'per-document' attributes, but are not allowed for 'intensive' 'MULTI-  
 967 ROW' attributes, such as mediumConsumed(171) and documentFormat(38)  
 968 which are specified to be 'per-job' attributes.

#### 969 3.3.5 Requested Objects and Attributes

970 A number of objects and attributes record requirements for the job.  
 971 Such object and attribute names end with the word 'Requested'. In the  
 972 interests of brevity, the phrase 'requested' SHALL mean: (1) requested  
 973 by the client (or intervening server) in the job submission protocol  
 974 and MAY also mean (2) embedded in the submitted document data, and/or  
 975 (3) defaulted by the recipient device or server with the same semantics  
 976 as if the requester had supplied, depending on implementation. Also if  
 977 a value is supplied by the job submission client, and the server/device  
 978 determines a better value, through processing or other means, the agent  
 979 MAY return that better value for such object and attribute.

## 980 3.3.6 Consumption Attributes

981 A number of objects and attributes record consumption. Such attribute  
982 names end with the word 'Completed' or 'Consumed'. If the job has not  
983 yet consumed what that resource is metering, the agent either: (1)  
984 SHALL return the value 0 or (2) SHALL not add this attribute to the  
985 jmAttributeTable until the consumption begins. In the interests of  
986 brevity, the semantics for 0 is specified once here and is not repeated  
987 for each consumption attribute specification and a DEFVAL of 0 is  
988 indicated.

## 989 3.3.7 Index Value Attributes

990 A number of attributes are indexes in other tables. Such attribute  
991 names end with the word 'Index'. If the agent has not (yet) assigned  
992 an index value for a particular index attribute for a job, the agent  
993 SHALL either: (1) return the value 0 or (2) not add this attribute to  
994 the jmAttributeTable until the index value is assigned. In the  
995 interests of brevity, the semantics for 0 is specified once here and is  
996 not repeated for each index attribute specification and a DEFVAL of 0  
997 is indicated.

## 998 3.4 Monitoring Job Progress

999 There are a number of objects and attributes for monitoring the  
1000 progress of a job. These objects and attributes count the number of K  
1001 octets, impressions, sheets, and pages requested or completed. For  
1002 impressions and sheets, "completed" SHALL mean stacked, unless the  
1003 implementation is unable to detect when each sheet is stacked, in which  
1004 case stacked is approximated when processing of each sheet completes.  
1005 There are objects and attributes for the overall job and for the  
1006 current copy of the document currently being stacked. For the latter,  
1007 the rate at which the various objects and attributes count depends on  
1008 the sheet and document collation of the job.

1009 Job Collation included sheet collation and document collation. Sheet  
1010 collation is defined to be the ordering of sheets within a document  
1011 copy. Document collation is defined to be ordering of document copies  
1012 within a multi-document job. There are three types of job collation  
1013 (see terminology definitions in Section 2):

1014 1. ~~u~~Uncollated-Sheets(3) - No collation of the sheets within each  
1015 document copy, i.e., each sheet of a document that is to  
1016 produce multiple copies is replicated before the next sheet in  
1017 the document is processed and stacked. If the device has an  
1018 output bin collator, ~~the~~ uncollated-~~s~~Sheets(3) value may  
1019 actually produce collated sheets as far as the user is  
1020 concerned (in the output bins). However, when the job  
1021 collation is ~~the~~ 'uncollated-~~s~~Sheets(3)' value, job progress is  
1022 indistinguishable to a monitoring application between a device  
1023 that has an output bin collator and one that does not.

1024 2. ~~c~~Collated-Documents(4) - Collation of the sheets within each  
 1025 document copy is performed within the printing device by making  
 1026 multiple passes over either the source or an intermediate  
 1027 representation of the document. In addition, when there are  
 1028 multiple documents per job, the i'th copy of each document is  
 1029 stacked before the j'th copy of each document, i.e., the  
 1030 documents are collated within each job copy. For example, if a  
 1031 job is submitted with documents, A and B, the job is made  
 1032 available to the end user as: A, B, A, B, .... The ~~c~~'Collated  
 1033 Documents(4)' value corresponds to the IPP [ipp-model]  
 1034 'separate-documents-collated-copies' value of the "multiple-  
 1035 document-handling" attribute.  
 1036

1037 If jobCopiesRequested or documentCopiesRequested = 1, then  
 1038 jobCollationType is defined as 4.

1039 3. ~~u~~Uncollated-Documents(5) - Collation of the sheets within each  
 1040 document copy is performed within the printing device by making  
 1041 multiple passes over either the source or an intermediate  
 1042 representation of the document. In addition, when there are  
 1043 multiple documents per job, all copies of the first document in  
 1044 the job are stacked before the any copied of the next document  
 1045 in the job, i.e., the documents are uncollated within the job.  
 1046 For example, if a job is submitted with documents, A and B, the  
 1047 job is mad available to the end user as: A, A, ..., B, B, ....  
 1048 The '~~u~~Uncollated-Documents(5)' value corresponds to the IPP  
 1049 [ipp-model] 'separate-documents-uncollated-copies' value of the  
 1050 "multiple-document-handling" attribute.

1051 Consider the following four variables that are used to monitor the  
 1052 progress of a job's impressions:

- 1053 1. jmJobImpressionsCompleted - counts the total number of  
 1054 impressions stacked for the job
- 1055 2. impressionsCompletedCurrentCopy - counts the number of  
 1056 impressions stacked for the current document copy
- 1057 3. sheetCompletedCopyNumber - identifies the number of the copy  
 1058 for the current document being stacked where the first copy is  
 1059 1.
- 1060 4. sheetCompletedDocumentNumber - identifies the current document  
 1061 within the job that is being stacked where the first document  
 1062 in a job is 1. NOTE: this attribute SHOULD NOT be implemented  
 1063 for implementations that only support one document per job.

1064 For each of the three types of job collation, a job with three copies  
 1065 of two documents (1, 2), where each document consists of 3 impressions,  
 1066 the four variables have the following values as each sheet is stacked  
 1067 for one-sided printing:

1068 Job Collation Type = ~~u~~Uncollated-Sheets(3)

1069

jmJobImpressions Completed	Impressions CompletedCurrent Copy	sheetCompleted CopyNumber	sheetCompleted DocumentNumber
0	0	0	0
1	1	1	1
2	1	2	1
3	1	3	1
4	2	1	1
5	2	2	1
6	2	3	1
7	3	1	1
8	3	2	1
9	3	3	1
10	1	1	2
11	1	2	2
12	1	3	2
13	2	1	2
14	2	2	2
15	2	3	2
16	3	1	2
17	3	2	2
18	3	3	2

1070

1071 Job Collation Type = ~~c~~ollated-~~D~~ocuments(4)

1072

JmJobImpressions Completed	Impressions CompletedCurrent Copy	sheetCompleted CopyNumber	sheetCompleted DocumentNumber
0	0	0	0
1	1	1	1
2	2	1	1
3	3	1	1
4	1	1	2
5	2	1	2
6	3	1	2
7	1	2	1
8	2	2	1
9	3	2	1
10	1	2	2
11	2	2	2
12	3	2	2
13	1	3	1
14	2	3	1
15	3	3	1
16	1	3	2
17	2	3	2
18	3	3	2

1073

1074 Job Collation Type = ~~u~~ncollated-~~Documents~~(5)  
1075

jmJobImpressions Completed	Impressions CompletedCurrent Copy	sheetCompleted CopyNumber	sheetCompleted DocumentNumber
0	0	0	0
1	1	1	1
2	2	1	1
3	3	1	1
4	1	2	1
5	2	2	1
6	3	2	1
7	1	3	1
8	2	3	1
9	3	3	1
10	1	1	2
11	2	1	2
12	3	1	2
13	1	2	2
14	2	2	2
15	3	2	2
16	1	3	2
17	2	3	2
18	3	3	2

1076

## 1077 3.5 Job Identification

1078 There are a number of attributes that permit a user, operator or system  
1079 administrator to identify jobs of interest, such as jobURI, jobName,  
1080 jobOriginatingHost, etc. In addition, there is a jmJobSubmissionID  
1081 object that is a text string table index. Being a table index allows a  
1082 monitoring application to quickly locate and identify a particular job  
1083 of interest that was submitted from a particular client by the user  
1084 invoking the monitoring application without having to scan the entire  
1085 job table. The Job Monitoring MIB needs to provide for identification  
1086 of the job at both sides of the job submission process. The primary  
1087 identification point is the client side. The jmJobSubmissionID allows  
1088 the monitoring application to identify the job of interest from all the  
1089 jobs currently "known" by the server or device. The value of  
1090 jmJobSubmissionID can be assigned by either the client's local system  
1091 or a downstream server or device. The point of assignment depends on  
1092 the job submission protocol in use.

1093 The server/device-side identifier, called the jmJobIndex object, SHALL  
1094 be assigned by the SNMP Job Monitoring MIB agent when the server or  
1095 device accepts the jobs from submitting clients. The jmJobIndex object  
1096 allows the interested party to obtain all objects desired that relate  
1097 to a particular job. See Section 3.2, entitled 'The Job Tables and the

1098 Oldest Active and Newest Active Indexes' for the specification of how  
1099 the agent SHALL assign the jmJobIndex values.

1100 The MIB provides a mapping table that maps each jmJobSubmissionID value  
1101 to a corresponding jmJobIndex value generated by the agent, so that an  
1102 application can determine the correct value for the jmJobIndex value  
1103 for the job of interest in a single Get operation, given the Job  
1104 Submission ID. See the jmJobIDGroup.

1105 In some configurations there may be more than one application program  
1106 that monitors the same job when the job passes from one network entity  
1107 to another when it is submitted. See configuration 3. When there are  
1108 multiple job submission IDs, each entity MAY supply an appropriate  
1109 jmJobSubmissionID value. In this case there would be a separate entry  
1110 in the jmJobSubmissionID table, one for each jmJobSubmissionID. All  
1111 entries would map to the same jmJobIndex that contains the job data.  
1112 When the job is deleted, it is up to the agent to remove all entries  
1113 that point to the job from the jmJobSubmissionID table as well.

1114 The jobName attribute provides a name that the user supplies as a job  
1115 attribute with the job. The jobName attribute is not necessarily  
1116 unique, even for one user, let alone across users.

1117 3.6 Internationalization Considerations

1118 This section describes the internationalization considerations included  
1119 in this MIB.

1120 3.6.1 Text generated by the server or device

1121 There are a few objects and attributes generated by the server or  
1122 device that SHALL be represented using the Universal Multiple-Octet  
1123 Coded Character Set (UCS) [ISO-10646]. These objects and attributes  
1124 are always supplied (if implemented) by the agent, not by the job  
1125 submitting client:

- 1126 1. jmGeneralJobSetName object
- 1127 2. processingMessage(6) attribute
- 1128 3. physicalDevice(32) (name value) attribute

1129 The character encoding scheme for representing these objects and  
1130 attributes SHALL be UTF-8 as recommended by RFC 2130 [RFC 2130] and the  
1131 "IETF Policy on Character Sets and Language" [char-set policy]. The  
1132 'JmUTF8StringTC' textual convention is used to indicate UTF-8 text  
1133 strings.

1134 NOTE - For strings in 7-bit US-ASCII, there is no impact since the UTF-  
1135 8 representation of 7-bit ASCII is identical to the US-ASCII [US-ASCII]  
1136 encoding.

1137 The text contained in the processingMessage(6) attribute is generated  
1138 by the server/device. The natural language for the  
1139 processingMessage(6) attribute is identified by the



1140 processingMessageNaturalLangTag(7) attribute. The  
1141 processingMessageNaturalLangTag(7) attribute uses the  
1142 JmNaturalLanguageTagTC textual convention which SHALL conform to the  
1143 language tag mechanism specified in RFC 1766 [RFC-1766]. The  
1144 JmNaturalLanguageTagTC value is the same as the IPP [IPP-model]  
1145 'naturalLanguage' attribute syntax. RFC 1766 specifies that a US-ASCII  
1146 string consisting of the natural language followed by an optional  
1147 country field. Both fields use the same two-character codes from ISO  
1148 639 [ISO-639] and ISO 3166 [ISO-3166], respectively, that are used in  
1149 the Printer MIB for identifying language and country.

1150 Examples of the values of the processingMessageNaturalLangTag(7)  
1151 attribute include:

- 1152 1. 'en' for English
- 1153 2. 'en-us' for US English
- 1154 3. 'fr' for French
- 1155 4. 'de' for German

### 1156 3.6.2 Text supplied by the job submitter

1157 All of the objects and attributes represented by the 'JmJobStringTC'  
1158 textual-convention are either (1) supplied in the job submission  
1159 protocol by the client that submits the job to the server or device or  
1160 (2) are defaulted by the server or device if the job submitting client  
1161 does not supply values. The agent SHALL represent these objects and  
1162 attributes in the MIB either (1) in the coded character set as they  
1163 were submitted or (2) MAY convert the coded character set to another  
1164 coded character set or encoding scheme. In any case, the resulting  
1165 coded character set representation SHOULD be UTF-8 [UTF-8], but SHALL  
1166 be one in which the code positions from 0 to 31 SHALL not be used, 32  
1167 to 127 SHALL be US-ASCII [US-ASCII], 127 SHALL be unused, and the  
1168 remaining code positions 128 to 255 SHALL represent single-byte or  
1169 multi-byte graphic characters structured according to ISO 2022 [ISO  
1170 2022] or SHALL be unused.

1171 The coded character set SHALL be one of the ones registered with IANA  
1172 [IANA] and SHALL be identified by the jobCodedCharSet attribute in the  
1173 jmJobAttributeTable for the job. If the agent does not know what coded  
1174 character set was used by the job submitting client, the agent SHALL  
1175 either (1) return the 'unknown(2)' value for the jobCodedCharSet  
1176 attribute or (2) not return the jobCodedCharSet attribute for the job.

1177 Examples of coded character sets which meet this criteria for use as  
1178 the value of the jobCodedCharSet job attribute are: US-ASCII [US-  
1179 ASCII], ISO 8859-1 (Latin-1) [ISO 8859-1], any ISO 8859-n, HP Roman8,  
1180 IBM Code Page 850, Windows Default 8-bit set, UTF-8 [UTF-8], US-ASCII  
1181 plus JIS X0208-1990 Japanese [JIS X0208], US-ASCII plus GB2312-1980 PRC  
1182 Chinese [GB2312]. See the IANA registry of coded character sets [IANA  
1183 charsets].

1184 Examples of coded character sets which do not meet this criteria are:  
1185 national 7-bit sets conforming to ISO 646 (except US-ASCII), EBCDIC,

1186 and ISO 10646 (Unicode) [ISO-10646]. In order to represent Unicode  
1187 characters, the UTF-8 [UTF-8] encoding scheme SHALL be used which has  
1188 been assigned the MIBenum value of '106' by IANA.

1189 The jobCodedCharSet attribute uses the imported 'CodedCharSet' textual-  
1190 convention from the Printer MIB [printmib].

1191 The natural language for attributes represented by the textual-  
1192 convention JmJobStringTC SHALL be identified either (1) by the  
1193 jobNaturalLanguageTag(9) attribute or SHALL be keywords in US-English  
1194 (as in IPP). A monitoring application SHOULD attempt to localize  
1195 keywords into the language of the user by means of some lookup  
1196 mechanism. If the keyword value is not known to the monitoring  
1197 application, the monitoring application SHOULD assume that the value is  
1198 in the natural language specified by the job's jobNaturalLanguageTag(9)  
1199 attribute and SHOULD present the value to its user as is. The  
1200 jobNaturalLanguageTag(9) attribute value SHALL have the same syntax and  
1201 semantics as the processingMessageNaturalLangTag(7) attribute, except  
1202 that the jobNaturalLanguageTag(9) attribute identifies the natural  
1203 language of attributes supplied by the job submitter instead of the  
1204 natural language of the processingMessage(6) attribute. See Section  
1205 3.6.1.

1206 3.6.3 'DateAndTime' for representing the date and time

1207 This MIB also contains objects that are represented using the  
1208 DateAndTime textual convention from SMIV2 [SMIV2-TC]. The job  
1209 management application SHALL display such objects in the locale of the  
1210 user running the monitoring application.

1211 3.7 IANA and PWG Registration Considerations

1212 This MIB does not require any additional registration schemes for IANA,  
1213 but does depend on registration schemes that other Internet standards  
1214 track specifications have set up. The names of these IANA registration  
1215 assignments under the /in-notes/iana/assignments/ path:

- 1216 1. printer-language-numbers - used as enums in the documentFormat(38)  
1217 attribute
- 1218 2. media-types - uses as keywords in the documentFormat(38) attribute
- 1219 3. character-sets - used as enums in the jobCodedCharSet(8) attribute

1220 ~~During the development of this standard, the Printer Working Group~~  
1221 ~~(PWG) will register additional enums while the standard is in the~~  
1222 ~~proposed and draft states according to the procedures described in this~~  
1223 ~~section.~~—The Printer Working Group (PWG) will handle registration of  
1224 additional enums after approving this standard, according to the  
1225 procedures described in this section:

## 1226 3.7.1 PWG Registration of enums

1227 This specification uses textual conventions to define enumerated values  
1228 (enums) and bit values. Enumerations (enums) and bit values are sets  
1229 of symbolic values defined for use with one or more objects or  
1230 attributes. All enumeration sets and bit value sets are assigned a  
1231 symbolic data type name (textual convention). As a convention the  
1232 symbolic name ends in "TC" for textual convention. These enumerations  
1233 are defined at the beginning of the MIB module specification.

1234 The PWG has defined several type of enumerations for use in the Job  
1235 Monitoring MIB and the Printer MIB[print-mib]. These types differ in  
1236 the method employed to control the addition of new enumerations.  
1237 Throughout this document, references to "type n enum", where n can be  
1238 1, 2 or 3 can be found in the various tables. The definitions of these  
1239 types of enumerations are:

## 1240 3.7.1.1 Type 1 enumerations

1241 Type 1 enumeration: All the values are defined in the Job Monitoring  
1242 MIB specification (RFC for the Job Monitoring MIB). Additional  
1243 enumerated values require a new RFC.

1244 There are no type 1 enums in the current draft.

## 1245 3.7.1.2 Type 2 enumerations

1246 Type 2 enumeration: An initial set of values are defined in the Job  
1247 Monitoring MIB specification. Additional enumerated values are  
1248 registered with the PWG.

1249 The following type 2 enums are contained in the current draft :

- 1250 1. JmUTF8StringTC
- 1251 2. JmJobStringTC
- 1252 3. JmNaturalLanguageTagTC
- 1253 4. JmTimeStampTC
- 1254 5. JmFinishingTC [same enum values as IPP "finishing" attribute]
- 1255 6. JmPrintQualityTC [same enum values as IPP "print-quality"  
1256 attribute]
- 1257 7. JmTonerEconomyTC
- 1258 8. JmMediumTypeTC
- 1259 9. JmJobSubmissionIDTypeTC
- 1260 10. JmJobCollationTypeTC
- 1261 11. JmJobStateTC [same enum values as IPP "job-state" attribute]
- 1262 12. JmAttributeTypeTC

1263 For those textual conventions that have the same enum values as the  
1264 indicated IPP Job attribute SHALL be simultaneously registered by the  
1265 PWG for use with IPP [ipp-model] and the Job Monitoring MIB.

## 1266 3.7.1.3 Type 3 enumeration

1267 Type 3 enumeration: An initial set of values are defined in the Job  
1268 Monitoring MIB specification. Additional enumerated values are  
1269 registered through the PWG without PWG review.

1270 There are no type 3 enums in the current draft.

## 1271 3.7.2 PWG Registration of type 2 bit values

1272 This draft contains the following type 2 bit value textual-conventions:

- 1273 1. JmJobServiceTypesTC
- 1274 2. JmJobStateReasons1TC
- 1275 3. JmJobStateReasons2TC
- 1276 4. JmJobStateReasons3TC
- 1277 5. JmJobStateReasons4TC

1278 These textual-conventions are defined as bits in an Integer so that  
1279 they can be used with SNMPv1 SMI. The jobStateReasonsN (N=1..4)  
1280 attributes are defined as bit values using the corresponding  
1281 JmJobStateReasonsMTC textual-conventions.

1282 The registration of JmJobServiceTypesTC and JmJobStateReasonsMTC bit  
1283 values SHALL follow the procedures for a type 2 enum as specified in  
1284 Section 3.7.1.2.

## 1285 3.7.3 PWG Registration of Job Submission Id Formats

1286 In addition to enums and bit values, this specification assigns a  
1287 single ASCII digit or letter to various job submission ID formats. See  
1288 the JmJobSubmissionIDTypeTC textual-convention and the object. The  
1289 registration of JobSubmissionID format numbers SHALL follow the  
1290 procedures for a type 2 enum as specified in Section 3.7.1.2.

## 1291 3.7.4 PWG Registration of MIME types/sub-types for document-formats

1292 The documentFormat(38) attribute has MIME type/sub-type values for  
1293 indicating document formats which IANA registers as "media type" names.  
1294 The values of the documentFormat(38) attribute are the same as the  
1295 corresponding Internet Printing Protocol (IPP) "document-format" Job  
1296 attribute values [ipp-model].

## 1297 3.8 Security Considerations

## 1298 3.8.1 Read-Write objects

1299 All objects are read-only, greatly simplifying the security  
1300 considerations. If another MIB augments this MIB, that MIB might  
1301 accept SNMP Write operations to objects in that MIB whose effect is to  
1302 modify the values of read-only objects in this MIB. However, that MIB  
1303 SHALL have to support the required access control in order to achieve  
1304 security, not this MIB.

## 1305 3.8.2 Read-Only Objects In Other User's Jobs

1306 The security policy of some sites MAY be that unprivileged users can  
1307 only get the objects from jobs that they submitted, plus a few minimal  
1308 objects from other jobs, such as the jmJobKOctetsPerCopyRequested and  
1309 jmJobKOctetsProcessed objects, so that a user can tell how busy a  
1310 printer is. Other sites MAY allow all unprivileged users to see all  
1311 objects of all jobs. This MIB does not require, nor does it specify  
1312 how, such restrictions would be implemented. A monitoring application  
1313 SHOULD enforce the site security policy with respect to returning  
1314 information to an unprivileged end user that is using the monitoring  
1315 application to monitor jobs that do not belong to that user, i.e., the  
1316 jmJobOwner object in the jmJobTable does not match the user's user  
1317 name.

1318 An operator is a privileged user that would be able to see all objects  
1319 of all jobs, independent of the policy for unprivileged users.

## 1320 3.9 Notifications

1321 This MIB does not specify any notifications. For simplicity,  
1322 management applications are expected to poll for status. The  
1323 jmGeneralJobPersistence and jmGeneralAttributePersistence objects  
1324 assist an application to determine the polling rate. The resulting  
1325 network traffic is not expected to be significant.

## 1326 4. MIB specification

1327 The following pages constitute the actual Job Monitoring MIB.

```
1328 Job-Monitoring-MIB DEFINITIONS ::= BEGIN
1329
1330 IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE, enterprises,
    Integer32                                FROM SNMPv2-SMI
    TEXTUAL-CONVENTION                       FROM SNMPv2-TC
    MODULE-COMPLIANCE, OBJECT-GROUP         FROM SNMPv2-CONF;
    -- The following textual-conventions are needed to implement
    -- certain attributes, but are not needed to compile this MIB.
    -- They are provided here for convenience:
    -- hrDeviceIndex                         FROM HOST-RESOURCES-MIB
    -- DateAndTime                          FROM SNMPv2-TC
    -- PrtInterpreterLangFamilyTC,
    -- CodedCharSet                         FROM Printer-MIB

1331
1332 -- Use the enterprises arc assigned to the PWG which is pwg(2699).
1333 -- Group all PWG mibs under mibs(1).
1334 --- Assign the first value: jobmonMIB(1) immediately under pwg(2669).
1335
1336 jobmonMIB MODULE-IDENTITY
1337     LAST-UPDATED "9801130000Z9802030000Z"
1338     ORGANIZATION "Printer Working Group (PWG)"
1339     CONTACT-INFO
1340         "Tom Hastings
1341         Postal: Xerox Corp.
1342                 Mail stop ESAE-231
1343                 701 S. Aviation Blvd.
1344                 El Segundo, CA 90245
1345
1346         Tel:      (301)333-6413
1347         Fax:      (301)333-5514
1348         E-mail:   hastings@cpl0.es.xerox.com
1349
1350         Send questions and comments to the Printer Working Group (PWG)
1351         using the Job Monitoring Project (JMP) Mailing List:
1352         jmp@pwg.org
1353
1354         For further information, including how to subscribe to the
1355         jmp mailing list, access the PWG web page under 'JMP':
1356
1357         http://www.pwg.org/
1358
1359         Implementers of this specification are encouraged to join the
1360         jmp mailing list in order to participate in discussions on any
1361         clarifications needed and registration proposals being reviewed
1362         in order to achieve consensus."
1363     DESCRIPTION
1364         "The MIB module for monitoring job in servers, printers, and
1365         other devices.
1366
1367         Version: 1.0"
1368     ::= { enterprises pwg(2699) mibs(1) jobmonMIB(1) }
```

```
1369
1370 -- Textual conventions for this MIB module
1371
1372 JmUTF8StringTC ::= TEXTUAL-CONVENTION
1373     DISPLAY-HINT "255a"
1374     STATUS      current
1375     DESCRIPTION
1376         "To facilitate internationalization, this TC represents
1377         information taken from the ISO/IEC IS 10646-1 character set,
1378         encoded as an octet string using the UTF-8 character encoding
1379         scheme."
1380     REFERENCE
1381         "See section 3.6.1, entitled: 'Text generated by the server or
1382         device'."
1383     SYNTAX      OCTET STRING (SIZE (0..63))
1384
1385
1386
1387
1388 JmJobStringTC ::= TEXTUAL-CONVENTION
1389     STATUS      current
1390     DESCRIPTION
1391         "To facilitate internationalization, this TC represents
1392         information using any coded character set registered by IANA as
1393         specified in section 3.7. While it is recommended that the
1394         coded character set be UTF-8 [UTF-8], the actual coded
1395         character set SHALL be indicated by the value of the
1396         jobCodedCharSet(8) attribute for the job."
1397     REFERENCE
1398         "See section 3.6.2, entitled: 'Text supplied by the job
1399         submitter'."
1400     SYNTAX      OCTET STRING (SIZE (0..63))
1401
1402
1403
1404
1405 JmNaturalLanguageTagTC ::= TEXTUAL-CONVENTION
1406     STATUS      current
1407     DESCRIPTION
1408         "An IETF RFC 1766-compliant 'language tag', with zero or more
1409         sub-tags that identify a natural language. While RFC 1766
1410         specifies that the US-ASCII values are case-insensitive, this
1411         MIB specification requires that all characters SHALL be lower
1412         case in order to simplify comparing by management
1413         applications."
1414     REFERENCE
1415         "See section 3.6.1, entitled: 'Text generated by the server or
1416         device' and section 3.6.2, entitled: 'Text supplied by the job
1417         submitter'."
1418     SYNTAX      OCTET STRING (SIZE (0..63))
1419
1420
```



```
1421 JmTimeStampTC ::= TEXTUAL-CONVENTION
1422     STATUS      current
1423     DESCRIPTION
1424         "The simple time at which an event took place.  The units SHALL
1425         be in seconds since the system was booted.
1426
1427         NOTE - JmTimeStampTC is defined in units of seconds, rather
1428         than 100ths of seconds, so as to be simpler for agents to
1429         implement (even if they have to implement the 100ths of a
1430         second to comply with implementing sysUpTime in MIB-II[mib-
1431         II].)
1432
1433         NOTE - JmTimeStampTC is defined as an Integer32 so that it can
1434         be used as a value of an attribute, i.e., as a value of the
1435         jmAttributeValueAsInteger object.  The TimeStamp textual-
1436         convention defined in SNMPv2-TC [SMIv2-TC] is defined as an
1437         APPLICATION 3 IMPLICIT INTEGER tag, not an Integer32 which is
1438         defined in SNMPv2-SMI [SMIv2-TC] as UNIVERSAL 2 IMPLICIT
1439         INTEGER, so cannot be used in this MIB as one of the values of
1440         jmAttributeValueAsInteger."
1441     SYNTAX      INTEGER (0..2147483647)
1442
1443
1444
1445
1446 JmJobSourcePlatformTypeTC ::= TEXTUAL-CONVENTION
1447     STATUS      current
1448     DESCRIPTION
1449         "The source platform type that can submit jobs to servers or
1450         devices in any of the 3 configurations."
1451     REFERENCE
1452         "This is a type 2 enumeration.  See Section 3.7.1.2.  See also
1453         IANA operating-system-names registry."
1454     SYNTAX      INTEGER {
1455         other(1),
1456         unknown(2),
1457         sptUNIX(3),           -- UNIX
1458         sptOS2(4),           -- OS/2
1459         sptPCDOS(5),         -- DOS
1460         sptNT(6),           -- NT
1461         sptMVS(7),          -- MVS
1462         sptVM(8),           -- VM
1463         sptOS400(9),        -- OS/400
1464         sptVMS(10),         -- VMS
1465         sptWindows(11),     -- Windows
1466         sptNetWare(12)      -- NetWare
1467     }
```



```
1457
1458 JmFinishingTC ::= TEXTUAL-CONVENTION
1459     STATUS      current
1460     DESCRIPTION
1461         "The type of finishing operation.
1462
1463         These values are the same as the enum values of the IPP
1464         'finishings' attribute.  See Section 3.7.1.2.
1465
1466         other(1),
1467             Some other finishing operation besides one of the specified
1468             or registered values.
1469
1470         unknown(2),
1471             The finishing is unknown.
1472
1473         none(3),
1474             Perform no finishing.
1475
1476         staple(4),
1477             Bind the document(s) with one or more staples. The exact
1478             number and placement of the staples is site-defined.
1479
1480         punch(5),
1481             This value indicates that holes are required in the
1482             finished document. The exact number and placement of the
1483             holes is site-defined. The punch specification MAY be
1484             satisfied (in a site- and implementation-specific manner)
1485             either by drilling/punching, or by substituting pre-drilled
1486             media.
1487
1488         cover(6),
1489             This value is specified when it is desired to select a non-
1490             printed (or pre-printed) cover for the document. This does
1491             not supplant the specification of a printed cover (on cover
1492             stock medium) by the document itself.
1493
1494         bind(7)
1495             This value indicates that a binding is to be applied to the
1496             document; the type and placement of the binding is product-
1497             specific."
1498     REFERENCE
1499         "This is a type 2 enumeration.  See Section 3.7.1.2."
1500     SYNTAX      INTEGER {
1501         other(1),
1502         unknown(2),
1503         none(3),
1504         staple(4),
1505         punch(5),
1506         cover(6),
1507         bind(7)
1508     }
```

```
1509
1510
1511 JmPrintQualityTC ::= TEXTUAL-CONVENTION
1512     STATUS      current
1513     DESCRIPTION
1514         "Print quality settings.
1515
1516         These values are the same as the enum values of the IPP 'print-
1517         quality' attribute.  See Section 3.7.1.2."
1518     REFERENCE
1519         "This is a type 2 enumeration.  See Section 3.7.1.2."
1520     SYNTAX      INTEGER {
1521         other(1),      -- Not one of the specified or registered
1522                       -- values.
1523         unknown(2),   -- The actual value is unknown.
1524         draft(3),     -- Lowest quality available on the printer.
1525         normal(4),    -- Normal or intermediate quality on the
1526                       -- printer.
1527         high(5)       -- Highest quality available on the printer.
1528     }
1529
1530
1531 JmPrinterResolutionTC ::= TEXTUAL-CONVENTION
1532     STATUS      current
1533     DESCRIPTION
1534         "Printer resolutions.
1535
1536         Nine octets consisting of two 4-octet SIGNED-INTEGERS followed
1537         by a SIGNED-BYTE.  The values are the same as those specified
1538         in the Printer MIB [printmib].  The first SIGNED-INTEGER
1539         contains the value of prtMarkerAddressabilityXFeedDir.  The
1540         second SIGNED-INTEGER contains the value of
1541         prtMarkerAddressabilityFeedDir.  The SIGNED-BYTE contains the
1542         value of prtMarkerAddressabilityUnit.
1543
1544         Note: the latter value is either 3 (tenThousandsOfInches) or 4
1545         (micrometers) and the addressability is in 10,000 units of
1546         measure.  Thus the SIGNED-INTEGERS represent integral values in
1547         either dots-per-inch or dots-per-centimeter.
1548
1549         The syntax is the same as the IPP 'printer-resolution'
1550         attribute.  See Section 3.7.1.2."
1551     SYNTAX      OCTET STRING (SIZE(9))
```

```
1548
1549 JmTonerEconomyTC ::= TEXTUAL-CONVENTION
1550     STATUS      current
1551     DESCRIPTION
1552         "Toner economy settings."
1553     REFERENCE
1554         "This is a type 2 enumeration.  See Section 3.7.1.2."
1555     SYNTAX      INTEGER {
1556         unknown(2),      -- unknown.
1557         off(3),          -- Off. Normal. Use full toner.
1558         on(4)            -- On. Use less toner than normal.
1559     }

1560 JmBooleanTC ::= TEXTUAL-CONVENTION
1561     STATUS      current
1562     DESCRIPTION
1563         "Boolean true or false value."
1564     REFERENCE
1565         "This is a type 2 enumeration.  See Section 3.7.1.2."
1566     SYNTAX      INTEGER {
1567         unknown(2),      -- unknown.
1568         false(3),        -- FALSE.
1569         true(4)          -- TRUE.
1570     }

1571 JmMediumTypeTC ::= TEXTUAL-CONVENTION
1572     STATUS      current
1573     DESCRIPTION
1574         "Identifies the type of medium.
1575
1576         other(1),
1577             The type is neither one of the values listed in this
1578             specification nor a registered value.
1579
1580         unknown(2),
1581             The type is not known.
1582
1583         stationery(3),
1584             Separately cut sheets of an opaque material.
1585
1586         transparency(4),
1587             Separately cut sheets of a transparent material.
1588
1589         envelope(5),
1590             Envelopes that can be used for conventional mailing
1591             purposes.
```

```
1592
1593     envelopePlain(6),
1594         Envelopes that are not preprinted and have no windows.
1595
1596     envelopeWindow(7),
1597         Envelopes that have windows for addressing purposes.
1598
1599     continuousLong(8),
1600         Continuously connected sheets of an opaque material
1601         connected along the long edge.
1602
1603     continuousShort(9),
1604         Continuously connected sheets of an opaque material
1605         connected along the short edge.
1606
1607     tabStock(10),
1608         Media with tabs.
1609
1610     multiPartForm(11),
1611         Form medium composed of multiple layers not pre-attached to
1612         one another; each sheet MAY be drawn separately from an
1613         input source.
1614
1615     labels(12),
1616         Label-stock.
1617
1618     multiLayer(13)
1619         Form medium composed of multiple layers which are pre-
1620         attached to one another, e.g. for use with impact
1621         printers."
1622 REFERENCE
1623     "This is a type 2 enumeration. See Section 3.7.1.2. These
1624     enum values correspond to the keyword name strings of the
1625     prtInputMediaType object in the Printer MIB [print-mib]. There
1626     is no printer description attribute in IPP/1.0 that represents
1627     these values."
1628 SYNTAX     INTEGER {
1629     other(1),
1630     unknown(2),
1631     stationery(3),
1632     transparency(4),
1633     envelope(5),
1634     envelopePlain(6),
1635     envelopeWindow(7),
1636     continuousLong(8),
1637     continuousShort(9),
1638     tabStock(10),
1639     multiPartForm(11),
1640     labels(12),
1641     multiLayer(13)
1642     }
1643
```

```
1644
1645 JmJobCollationTypeTC ::= TEXTUAL-CONVENTION
1646     STATUS      current
1647     DESCRIPTION
1648         "This value is the type of job collation. Implementations that
1649         don't support multiple documents or don't support multiple
1650         copies SHALL NOT support the uncollatedDocuments(5) value."
1651     REFERENCE
1652         "This is a type 2 enumeration. See Section 3.7.1.2. See also
1653         Section 3.4, entitled 'Monitoring Job Progress'."
1654     SYNTAX      INTEGER {
1655         other(1),
1656         unknown(2),
1657         uncollatedSheets(3),      -- sheets within each document copy
1658                                   -- are not collated: 1 1 ..., 2 2 ...,
1659         collatedDocuments(4),    -- internal collated sheets,
1660                                   -- documents: A, B, A, B, ...
1661         uncollatedDocuments(5)  -- internal collated sheets,
1662                                   -- documents: A, A, ..., B, B, ...
1663     }
1664
1665
1666 JmJobSubmissionIDTypeTC ::= TEXTUAL-CONVENTION
1667     STATUS      current
1668     DESCRIPTION
1669         "Identifies the format type of a job submission ID.
1670
1671         Each job submission ID is a fixed-length, 48-octet printable
1672         US-ASCII [US-ASCII] coded character string containing no
1673         control characters, consisting of the following fields:
1674
1675         octet 1: The format letter identifying the format. The US-
1676         ASCII characters '0-9', 'A-Z', and 'a-z' are assigned in
1677         order giving 62 possible formats.
1678         octets 2-40: A 39-character, US-ASCII trailing SPACE filled
1679         field specified by the format letter, if the data is less
1680         than 39 ASCII characters.
1681         octets 41-48: A sequential or random US-ASCII number to make
1682         the ID quasi-unique.
1683
1684         If the client does not supply a job submission ID in the job
1685         submission protocol, then the agent SHALL assign a job
1686         submission ID using any of the standard formats that are
1687         reserved for the agent. Clients SHALL not use formats that are
1688         reserved for agents and agents SHALL NOT use formats that are
1689         reserved for clients, in order to reduce conflicts in ID
1690         generation. See the description for which formats are reserved
1691         for clients or for agents.
1692
1693         Registration of additional formats may be done following the
1694         procedures described in Section 3.7.3.
1695
```

1696 The format values defined at the time of completion of this  
1697 specification are:  
1698  
1699 Format  
1700 Letter Description  
1701 -----  
1702 '0' Job Owner generated by the server/device  
1703 octets 2-40: The last 39 bytes of the jmJobOwner object.  
1704 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
1705 assigned by the agent.  
1706 This format is reserved for agents.  
1707  
1708 NOTE - Clients wishing to use a job submission ID that  
1709 incorporates the job owner, SHALL use format '8', not  
1710 format '0'.  
1711  
1712 '1' Job Name  
1713 octets 2-40: The last 39 bytes of the jobName attribute.  
1714 octets 41-48: The US-ASCII 8-decimal-digit random number  
1715 assigned by the client.  
1716 This format is reserved for clients.  
1717  
1718 '2' Client MAC address  
1719 octets 2-40: The client MAC address: in hexadecimal with each  
1720 nibble of the 6 octet address being '0'-'9' or 'A' - 'F'  
1721 (uppercase only). Most significant octet first.  
1722 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
1723 assigned by the client.  
1724 This format is reserved for clients.  
1725  
1726 '3' Client URL  
1727 octets 2-40: The last 39 bytes of the client URL [URI-spec].  
1728 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
1729 assigned by the client.  
1730 This format is reserved for clients.  
1731  
1732 '4' Job URI  
1733 octets 2-40: The last 39 bytes of the URI [URI-spec] assigned  
1734 by the server or device to the job when the job was  
1735 submitted for processing.  
1736 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
1737 assigned by the agent.  
1738 This format is reserved for agents.  
1739  
1740 '5' POSIX User Number  
1741 octets 2-40: The last 39 bytes of a user number, such as POSIX  
1742 user number.  
1743 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
1744 assigned by the client.  
1745 This format is reserved for clients.  
1746

1747 '6' User Account Number  
1748 octets 2-40: The last 39 bytes of the user account number.  
1749 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
1750 assigned by the client.  
1751 This format is reserved for clients.  
1752  
1753 '7' DTMF Incoming FAX routing number  
1754 octets 2-40: The last 39 bytes of the DTMF incoming FAX  
1755 routing number.  
1756 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
1757 assigned by the client.  
1758 This format is reserved for clients.  
1759  
1760 '8' Job Owner supplied by the client  
1761 octets 2-40: The last 39 bytes of the job owner name (that the  
1762 agent returns in the jmJobOwner object).  
1763 octets 41-48: The US-ASCII 8-decimal-digit sequential number  
1764 assigned by the client.  
1765 This format is reserved for clients. See format '0' which is  
1766 reserved for agents.  
1767  
1768 '9' Host Name  
1769 octets 2-40: The last 39 bytes of the host name with trailing  
1770 SPACES that submitted the job to this server/device using a  
1771 protocol, such as LPD [RFC-1179] which includes the host  
1772 name in the job submission protocol.  
1773 octets 41-48: The US-ASCII 8-decimal-digit leading zero  
1774 representation of the job id generated by the submitting  
1775 server (configuration 3) or the client (configuration 1 and  
1776 2), such as in the LPD protocol.  
1777 This format is reserved for clients.  
1778  
1779 'A' AppleTalk Protocol  
1780 octets 2-40: Contains the AppleTalk printer name, with the  
1781 first character of the name in octet 2. AppleTalk printer  
1782 names are a maximum of 31 characters. Any unused portion  
1783 of this field shall be filled with spaces.  
1784 octets 41-48: '00000XXX', where 'XXX' is the 3-digit US-ASCII  
1785 decimal representation of the Connection Id.  
1786 This format is reserved for agents.  
1787  
1788 'B' NetWare PServer  
1789 octets 2-40: Contains the Directory Path Name as recorded by  
1790 the Novell File Server in the queue directory. If the  
1791 string is less than 40 octets, the left-most character in  
1792 the string shall appear in octet position 2. Otherwise,  
1793 only the last 39 bytes shall be included. Any unused  
1794 portion of this field shall be filled with spaces.  
1795 octets 41-48: '000XXXXX' The US-ASCII representation of the  
1796 Job Number as per the NetWare File Server Queue Management  
1797 Services.  
1798 This format is reserved for agents.



1799  
1800 'C' Server Message Block protocol (SMB)  
1801 octets 2-40: Contains a decimal (US-ASCII coded)  
1802 representation of the 16 bit SMB Tree Id field, which  
1803 uniquely identifies the connection that submitted the job  
1804 to the printer. The most significant digit of the numeric  
1805 string shall be placed in octet position 2. All unused  
1806 portions of this field shall be filled with spaces. The  
1807 SMB Tree Id has a maximum value of 65,535.  
1808 octets 41-48: The US-ASCII 8-decimal-digit leading zero  
1809 representation of the File Handle returned from the device  
1810 to the client in response to a Create Print File command.  
1811 This format is reserved for agents.  
1812  
1813 'D' Transport Independent Printer/System Interface (TIP/SI)  
1814 octets 2-40: Contains the Job Name from the Job Control-Start  
1815 Job (JC-SJ) command. If the Job Name portion is less than  
1816 40 octets, the left-most character in the string shall  
1817 appear in octet position 2. Any unused portion of this  
1818 field shall be filled with spaces. Otherwise, only the  
1819 last 39 bytes shall be included.  
1820 octets 41-48: The US-ASCII 8-decimal-digit leading zero  
1821 representation of the jmJobIndex assigned by the agent.  
1822 This format is reserved for agents, since the agent supplies  
1823 octets 41-48, though the client supplies the job name. See  
1824 format '1' reserved to clients to submit job name ids in  
1825 which they supply octets 41-48.  
1826  
1827 'E' IPDS on the MVS or VSE platform  
1828  
1829 octets 2-40: Contains bytes 2-27 of the XOH Define Group  
1830 Boundary Group ID triplet. Octet position 2 MUST carry the  
1831 value x'01'. Bytes 28-40 MUST be filled with spaces.  
1832 octets 41-48: The US-ASCII 8-decimal-digit leading zero  
1833 representation of the jmJobIndex assigned by the agent.  
1834 This format is reserved for agents, since the agent supplies  
1835 octets 41-48, though the client supplies the job name.  
1836  
1837 'F' IPDS on the VM platform  
1838 octets 2-40: Contains bytes 2-31 of the XOH Define Group  
1839 Boundary Group ID triplet. Octet position 2 MUST carry the  
1840 value x'02'. Bytes 32-40 MUST be filled with spaces.  
1841 octets 41-48: The US-ASCII 8-decimal-digit leading zero  
1842 representation of the jmJobIndex assigned by the agent.  
1843 This format is reserved for agents, since the agent supplies  
1844 octets 41-48, though the client supplies the file name.  
1845

1846 'G' IPDS on the OS/400 platform  
1847 octets 2-40: Contains bytes 2-36 of the XOH Define Group  
1848 Boundary Group ID triplet. Octet position 2 MUST carry the  
1849 value x'03'. Bytes 37-40 MUST be filled with spaces.  
1850 octets 41-48: The US-ASCII 8-decimal-digit leading zero  
1851 representation of the jmJobIndex assigned by the agent.  
1852 This format is reserved for agents, since the agent supplies  
1853 octets 41-48, though the client supplies the job name.  
1854

1855 NOTE - the job submission id is only intended to be unique  
1856 between a limited set of clients for a limited duration of  
1857 time, namely, for the life time of the job in the context of  
1858 the server or device that is processing the job. Some of the  
1859 formats include something that is unique per client and a  
1860 random number so that the same job submitted by the same client  
1861 will have a different job submission id. For other formats,  
1862 where part of the id is guaranteed to be unique for each  
1863 client, such as the MAC address or URL, a sequential number  
1864 SHOULD suffice for each client (and may be easier for each  
1865 client to manage). Therefore, the length of the job submission  
1866 id has been selected to reduce the probability of collision to  
1867 an extremely low number, but is not intended to be an absolute  
1868 guarantee of uniqueness. None-the-less, collisions are  
1869 remotely possible, but without bad consequences, since this MIB  
1870 is intended to be used only for monitoring jobs, not for  
1871 controlling and managing them."

1872 REFERENCE

1873 "This is like a type 2 enumeration. See section 3.7.3."

1874 SYNTAX OCTET STRING(SIZE(1)) -- ASCII '0'-'9', 'A'-'Z', 'a'-'z'

```

1875
1876 JmJobStateTC ::= TEXTUAL-CONVENTION
1877     STATUS      current
1878     DESCRIPTION
1879         "The current state of the job (pending, processing, completed,
1880         etc.).
1881
1882     The following figure shows the normal job state transitions:
1883
1884                                     +----> canceled(7)
1885                                     /
1886     +----> pending(3) -----> processing(5) -----+-----> completed(9)
1887     |           ^           |           ^           |           \
1888     --->+       |           |           |           |           +-----> aborted(8)
1889     |           v           |           v           |           /
1890     +----> pendingHeld(4)  processingStopped(6) ----+
1891

```

Figure 4 - Normal Job State Transitions

Normally a job progresses from left to right. Other state transitions are unlikely, but are not forbidden. Not shown are the transitions to the canceled state from the pending, pendingHeld, and processingStopped states.

Jobs in the pending, processing, and processingStopped states are called 'active', while jobs in the pendingHeld, canceled, aborted, and completed states are called 'inactive'. Jobs reach one of the three terminal states: completed, canceled, or aborted, *after* the jobs have completed all activity, and all MIB objects and attributes have reached their final values for the job.

These values are the same as the enum values of the IPP 'job-state' job attribute. See Section 3.7.1.2.

unknown(2),

The job state is *not* known, or its state is indeterminate.

pending(3),

The job is a candidate to start processing, but is not yet processing.

pendingHeld(4),

The job is not a candidate for processing for any number of reasons but will return to the pending state as soon as the reasons are no longer present. The job's jmJobStateReasons1 object and/or jobStateReasonsN (N=2..4) attributes SHALL indicate why the job is no longer a candidate for processing. The reasons are represented as bits in the jmJobStateReasons1 object and/or jobStateReasonsN (N=2..4) attributes. See the

1926 JmJobStateReasonsMTC (N=1..4) textual convention for the  
1927 specification of each reason.  
1928  
1929 processing(5),  
1930 One or more of:  
1931  
1932 1. the job is using, or is attempting to use, one or more  
1933 purely software processes that are analyzing, creating, or  
1934 interpreting a PDL, etc.,  
1935  
1936 2. the job is using, or is attempting to use, one or more  
1937 hardware devices that are interpreting a PDL, making marks  
1938 on a medium, and/or performing finishing, such as stapling,  
1939 etc.,  
1940  
1941 OR  
1942  
1943 3. (configuration 2) the server has made the job ready for  
1944 printing, but the output device is not yet printing it,  
1945 either because the job hasn't reached the output device or  
1946 because the job is queued in the output device or some  
1947 other spooler, awaiting the output device to print it.  
1948  
1949 When the job is in the processing state, the entire job  
1950 state includes the detailed status represented in the  
1951 device MIB indicated by the hrDeviceIndex value of the  
1952 job's physicalDevice attribute, if the agent implements  
1953 such a device MIB.  
1954  
1955 Implementations MAY, though they NEED NOT, include  
1956 additional values in the job's jmJobStateReasons1 object to  
1957 indicate the progress of the job, such as adding the  
1958 jobPrinting value to indicate when the device is actually  
1959 making marks on a medium and/or the processingToStopPoint  
1960 value to indicate that the server or device is in the  
1961 process of canceling or aborting the job.  
1962  
1963 processingStopped(6),  
1964 The job has stopped while processing for any number of  
1965 reasons and will return to the processing state as soon as  
1966 the reasons are no longer present.  
1967  
1968 The job's jmJobStateReasons1 object and/or the job's  
1969 jobStateReasonsN (N=2..4) attributes MAY indicate why the  
1970 job has stopped processing. For example, if the output  
1971 device is stopped, the deviceStopped value MAY be included  
1972 in the job's jmJobStateReasons1 object.  
1973  
1974 NOTE - When an output device is stopped, the device usually  
1975 indicates its condition in human readable form at the  
1976 device. The management application can obtain more  
1977 complete device status remotely by querying the appropriate

1978 device MIB using the job's deviceIndex attribute(s), if the  
 1979 agent implements such a device MIB  
 1980  
 1981 canceled(7),  
 1982 A client has canceled the job and the server or device has  
 1983 completed canceling the job AND all MIB objects and  
 1984 attributes have reached their final values for the job.  
 1985 While the server or device is canceling the job, the job's  
 1986 jmJobStateReasons1 object SHOULD contain the  
 1987 processingToStopPoint value and one of the canceledByUser,  
 1988 canceledByOperator, or canceledAtDevice values. The  
 1989 canceledByUser, canceledByOperator, or canceledAtDevice  
 1990 values remain while the job is in the canceled state.  
 1991  
 1992 aborted(8),  
 1993 The job has been aborted by the system, usually while the  
 1994 job was in the processing or processingStopped state and  
 1995 the server or device has completed aborting the job AND all  
 1996 MIB objects and attributes have reached their final values  
 1997 for the job. While the server or device is aborting the  
 1998 job, the job's jmJobStateReasons1 object MAY contain the  
 1999 processingToStopPoint and abortedBySystem values. If  
 2000 implemented, the abortedBySystem value SHALL remain while  
 2001 the job is in the aborted state.  
 2002  
 2003 completed(9)  
 2004 The job has completed successfully or with warnings or  
 2005 errors after processing and all of the media have been  
 2006 successfully stacked in the appropriate output bin(s) AND  
 2007 all MIB objects and attributes have reached their final  
 2008 values for the job. The job's jmJobStateReasons1 object  
 2009 SHOULD contain one of: completedSuccessfully,  
 2010 completedWithWarnings, or completedWithErrors values."  
 2011 REFERENCE  
 2012 "This is a type 2 enumeration. See Section 3.7.1.2."  
 2013 SYNTAX INTEGER {  
 2014 unknown(2),  
 2015 pending(3),  
 2016 pendingHeld(4),  
 2017 processing(5),  
 2018 processingStopped(6),  
 2019 canceled(7),  
 2020 aborted(8),  
 2021 completed(9)  
 2022 }

2023  
 2024 JmAttributeTypeTC ::= TEXTUAL-CONVENTION  
 2025       STATUS       current  
 2026       DESCRIPTION  
 2027           "The type of the attribute which identifies the attribute.  
 2028  
 2029       In the following definitions of the enums, each description  
 2030       indicates whether the useful value of the attribute SHALL be  
 2031       represented using the jmAttributeValueAsInteger or the  
 2032       jmAttributeValueAsOctets objects by the initial tag: 'INTEGER:'  
 2033       or 'OCTETS:', respectively.  
 2034  
 2035       Some attributes allow the agent implementer a choice of useful  
 2036       values of either an integer, an octets representation, or both,  
 2037       depending on implementation. These attributes are indicated  
 2038       with 'INTEGER:' AND/OR 'OCTETS:' tags.  
 2039  
 2040       A very few attributes require both objects at the same time to  
 2041       represent a pair of useful values (see mediumConsumed(171)).  
 2042       These attributes are indicated with 'INTEGER:' AND 'OCTETS:'  
 2043       tags. See the jmAttributeGroup for the descriptions of these  
 2044       two MANDATORY objects.  
 2045  
 2046       NOTE - The enum assignments are grouped logically with values  
 2047       assigned in groups of 20, so that additional values may be  
 2048       registered in the future and assigned a value that is part of  
 2049       their logical grouping.  
 2050  
 2051       Values in the range 2\*\*30 to 2\*\*31-1 are reserved for private  
 2052       or experimental usage. This range corresponds to the same  
 2053       range reserved in IPP. Implementers are warned that use of  
 2054       such values may conflict with other implementations.  
 2055       Implementers are encouraged to request registration of enum  
 2056       values following the procedures in Section 3.7.1.  
 2057  
 2058       NOTE: No attribute name exceeds 31 characters.  
 2059  
 2060       The standard attribute types defined at the time of completion  
 2061       of the specification are:

jmAttributeTypeIndex	Datatype
-----	-----
other(1),	Integer32 (-2..2147483647) AND/OR OCTET STRING(SIZE(0..63))
INTEGER: and/or OCTETS:	An attribute that is not in the list and/or that has not been approved and registered with the PWG.

2069  
 2070  
 2071

2072 ++++++  
2073 + Job State attributes  
2074 +  
2075 + The following attributes specify the state of a job.  
2076 ++++++  
2077  
2078 jobStateReasons2(3), JmJobStateReasons2TC  
2079 INTEGER: Additional information about the job's current  
2080 state that augments the jmJobState object. See the  
2081 description under the JmJobStateReasons1TC textual-  
2082 convention.  
2083  
2084 jobStateReasons3(4), JmJobStateReasons3TC  
2085 INTEGER: Additional information about the job's current  
2086 state that augments the jmJobState object. See the  
2087 description under JmJobStateReasons1TC textual-convention.  
2088  
2089 jobStateReasons4(5), JmJobStateReasons4TC  
2090 INTEGER: Additional information about the job's current  
2091 state that augments the jmJobState object. See the  
2092 description under JmJobStateReasons1TC textual-convention.  
2093  
2094 processingMessage(6), JmUTF8StringTC\_(SIZE(0..63))  
2095 OCTETS: MULTI-ROW: A coded character set message that is  
2096 generated by the server or device during the processing of  
2097 the job as a simple form of processing log to show progress  
2098 and any problems. The natural language of each value is  
2099 specified by the corresponding  
2100 processingMessageNaturalLangTag(7) value.  
2101  
2102 NOTE - This attribute is intended for such conditions as  
2103 interpreter messages, rather than being the printable form  
2104 of the jmJobState and jmJobStateReasons1 objects and  
2105 jobStateReasons2, jobStateReasons3, and jobStateReasons4  
2106 attributes. In order to produce a localized printable form  
2107 of these job state objects/attribute, a management  
2108 application SHOULD produce a message from their enum and  
2109 bit values.  
2110  
2111 NOTE - There is no job description attribute in IPP/1.0  
2112 that corresponds to this attribute and this attribute does  
2113 not correspond to the IPP/1.0 'job-state-message' job  
2114 description attribute, which is just a printable form of  
2115 the IPP 'job-state' and 'job-state-reasons' job attributes.  
2116  
2117 There is no restriction for the same message occurring in  
2118 multiple rows.  
2119



2120 processingMessageNaturalLangTag(7), OCTET STRING(SIZE(0..63))  
2121 OCTETS: MULTI-ROW: The natural language of the  
2122 corresponding processingMessage(6) attribute value. See  
2123 section 3.6.1, entitled 'Text generated by the server or  
2124 device'.

2125  
2126 If the agent does not know the natural language of the job  
2127 processing message, the agent SHALL either (1) return a  
2128 zero length string value for the  
2129 processingMessageNaturalLangTag(7) attribute or (2) not  
2130 return the processingMessageNaturalLangTag(7) attribute for  
2131 the job.

2132  
2133 There is no restriction for the same tag occurring in  
2134 multiple rows, since when this attribute is implemented, it  
2135 SHOULD have a value row for each corresponding  
2136 processingMessage(6) attribute value row.

2137  
2138 jobCodedCharSet(8), CodedCharSet  
2139 INTEGER: The MIBenum identifier of the coded character set  
2140 that the agent is using to represent coded character set  
2141 objects and attributes of type 'JmJobStringTC'. These  
2142 coded character set objects and attributes are either: (1)  
2143 supplied by the job submitting client or (2) defaulted by  
2144 the server or device when omitted by the job submitting  
2145 client. The agent SHALL represent these objects and  
2146 attributes in the MIB either (1) in the coded character set  
2147 as they were submitted or (2) MAY convert the coded  
2148 character set to another coded character set or encoding  
2149 scheme as identified by the jobCodedCharSet(8) attribute.  
2150 See section 3.6.2, entitled 'Text supplied by the job  
2151 submitter'.

2152  
2153 These MIBenum values are assigned by IANA [IANA-charsets]  
2154 when the coded character sets are registered. The coded  
2155 character set SHALL be one of the ones registered with IANA  
2156 [IANA] and the enum value uses the CodedCharSet textual-  
2157 convention from the Printer MIB. See the JmJobStringTC  
2158 textual-convention.

2159  
2160 If the agent does not know what coded character set was  
2161 used by the job submitting client, the agent SHALL either  
2162 (1) return the 'unknown(2)' value for the  
2163 jobCodedCharSet(8) attribute or (2) not return the  
2164 jobCodedCharSet(8) attribute for the job.

2165

2166           jobNaturalLanguageTag(9),                   OCTET STRING(SIZE(0..63))  
2167            OCTETS: The natural language of the job attributes supplied  
2168            by the job submitter or defaulted by the server or device  
2169            for the job, i.e., all objects and attributes represented  
2170            by the 'JmJobStringTC' textual-convention, such as jobName,  
2171            mediumRequested, etc. See Section 3.6.2, entitled 'Text  
2172            supplied by the job submitter'.  
2173  
2174            If the agent does not know what natural language was used  
2175            by the job submitting client, the agent SHALL either (1)  
2176            return a zero length string value for the  
2177            jobNaturalLanguageTag(9) attribute or (2) not return  
2178            jobNaturalLanguageTag(9) attribute for the job.  
2179  
2180  
2181            +++++  
2182            + Job Identification attributes  
2183            +  
2184            + The following attributes help an end user, a system  
2185            + operator, or an accounting program identify a job.  
2186            +++++  
2187  
2188  
2189  
2190            jobURI(20),                                 OCTET STRING(SIZE(0..63))  
2191            OCTETS: MULTI-ROW: The job's Universal Resource  
2192            Identifier (URI) [RFC-1738]. See IPP [ipp-model] for  
2193            example usage.  
2194  
2195            NOTE - The agent may be able to generate this value on each  
2196            SNMP Get operation from smaller values, rather than having  
2197            to store the entire URI.  
2198  
2199            If the URI exceeds 63 octets, the agent SHALL use multiple  
2200            values, with the next 63 octets coming in the second value,  
2201            etc.  
2202  
2203            NOTE - IPP [ipp-model] has a 1023-octet maximum length for  
2204            a URI, though the URI standard itself and HTTP/1.1 specify  
2205            no maximum length.  
2206  
2207            jobAccountName(21),                         OCTET STRING(SIZE(0..63))  
2208            OCTETS: Arbitrary binary information which MAY be coded  
2209            character set data or encrypted data supplied by the  
2210            submitting user for use by accounting services to allocate  
2211            or categorize charges for services provided, such as a  
2212            customer account name or number.  
2213  
2214            NOTE: This attribute NEED NOT be printable characters.  
2215

2216 serverAssignedJobName(22), JmJobStringTC\_(SIZE(0..63))  
2217 OCTETS: Configuration 3 only: The human readable string  
2218 name, number, or ID of the job as assigned by the server  
2219 that submitted the job to the device that the agent is  
2220 providing access to with this MIB.  
2221  
2222 NOTE - This attribute is intended for enabling a user to  
2223 find his/her job that a server submitted to a device when  
2224 either the client does not support the jmJobSubmissionID or  
2225 the server does not pass the jmJobSubmissionID through to  
2226 the device.  
2227

2228 jobName(23), JmJobStringTC\_(SIZE(0..63))  
2229 OCTETS: The human readable string name of the job as  
2230 assigned by the submitting user to help the user  
2231 distinguish between his/her various jobs. This name does  
2232 not need to be unique.  
2233  
2234 This attribute is intended for enabling a user or the  
2235 user's application to convey a job name that MAY be printed  
2236 on a start sheet, returned in a query result, or used in  
2237 notification or logging messages.  
2238  
2239 In order to assist users to find their jobs for job  
2240 submission protocols that don't supply a jmJobSubmissionID,  
2241 the agent SHOULD maintain the jobName attribute for the  
2242 time specified by the jmGeneralJobPersistence object,  
2243 rather than the (shorter) jmGeneralAttributePersistence  
2244 object.  
2245  
2246 If this attribute is not specified when the job is  
2247 submitted, no job name is assumed, but implementation  
2248 specific defaults are allowed, such as the value of the  
2249 documentName attribute of the first document in the job or  
2250 the fileName attribute of the first document in the job.  
2251  
2252 The jobName attribute is distinguished from the jobComment  
2253 attribute, in that the jobName attribute is intended to  
2254 permit the submitting user to distinguish between different  
2255 jobs that he/she has submitted. The jobComment attribute  
2256 is intended to be free form additional information that a  
2257 user might wish to use to communicate with himself/herself,  
2258 such as a reminder of what to do with the results or to  
2259 indicate a different set of input parameters were tried in  
2260 several different job submissions.  
2261

2262           jobServiceTypes(24),                    JmJobServiceTypesTC  
2263            INTEGER: Specifies the type(s) of service to which the job  
2264            has been submitted (print, fax, scan, etc.). The service  
2265            type is bit encoded with each job service type so that more  
2266            general and arbitrary services can be created, such as  
2267            services with more than one destination type, or ones with  
2268            only a source or only a destination. For example, a job  
2269            service might scan, faxOut, and print a single job. In  
2270            this case, three bits would be set in the jobServiceTypes  
2271            attribute, corresponding to the hexadecimal values: 0x8 +  
2272            0x20 + 0x4, respectively, yielding: 0x2C.  
2273  
2274            Whether this attribute is set from a job attribute supplied  
2275            by the job submission client or is set by the recipient job  
2276            submission server or device depends on the job submission  
2277            protocol. This attribute SHALL be implemented if the  
2278            server or device has other types in addition to or instead  
2279            of printing.  
2280  
2281            One of the purposes of this attribute is to permit a  
2282            requester to filter out jobs that are not of interest. For  
2283            example, a printer operator may only be interested in jobs  
2284            that include printing.  
2285  
2286            jobSourceChannelIndex(25),            Integer32\_(0..2147483647)  
2287            INTEGER: The index of the row in the associated Printer  
2288            MIB[print-mib] of the channel which is the source of the  
2289            print job.  
2290  
2291            jobSourcePlatformType(26),            JmJobSourcePlatformTypeTC  
2292            INTEGER: The source platform type of the immediate  
2293            upstream submitter that submitted the job to the server  
2294            (configuration 2) or device (configuration 1 and 3) to  
2295            which the agent is providing access. For configuration 1,  
2296            this is the type of the client that submitted the job to  
2297            the device; for configuration 2, this is the type of the  
2298            client that submitted the job to the server; and for  
2299            configuration 3, this is the type of the server that  
2300            submitted the job to the device.  
2301  
2302            submittingServerName(27),            JmJobStringTC\_(SIZE(0..63))  
2303            OCTETS: For configuration 3 only: The administrative name  
2304            of the server that submitted the job to the device.  
2305  
2306            submittingApplicationName(28),        JmJobStringTC\_(SIZE(0..63))  
2307            OCTETS: The name of the client application (not the server  
2308            in configuration 3) that submitted the job to the server or  
2309            device.  
2310

2311 jobOriginatingHost(29), JmJobStringTC\_ (SIZE(0..63))  
2312 OCTETS: The name of the client host (not the server host  
2313 name in configuration 3) that submitted the job to the  
2314 server or device.  
2315

2316 deviceNameRequested(30), JmJobStringTC\_ (SIZE(0..63))  
2317 OCTETS: The administratively defined coded character set  
2318 name of the target device requested by the submitting user.  
2319 For configuration 1, its value corresponds to the Printer  
2320 MIB[print-mib]: prtGeneralPrinterName object. For  
2321 configuration 2 and 3, its value is the name of the logical  
2322 or physical device that the user supplied to indicate to  
2323 the server on which device(s) they wanted the job to be  
2324 processed.  
2325

2326 queueNameRequested(31), JmJobStringTC\_ (SIZE(0..63))  
2327 OCTETS: The administratively defined coded character set  
2328 name of the target queue requested by the submitting user.  
2329 For configuration 1, its value corresponds to the queue in  
2330 the device for which the agent is providing access. For  
2331 configuration 2 and 3, its value is the name of the queue  
2332 that the user supplied to indicate to the server on which  
2333 device(s) they wanted the job to be processed.  
2334

2335 NOTE - typically an implementation SHOULD support either  
2336 the deviceNameRequested or queueNameRequested attribute,  
2337 but not both.  
2338

2339 physicalDevice(32), hrDeviceIndex  
2340 AND/OR  
2341 JmUTF8StringTC\_ (SIZE(0..63))  
2342 INTEGER: MULTI-ROW: The index of the physical device MIB  
2343 instance requested/used, such as the Printer MIB[print-  
2344 mib]. This value is an hrDeviceIndex value. See the Host  
2345 Resources MIB[hr-mib].  
2346

2347 AND/OR

2348

2349 OCTETS: MULTI-ROW: The name of the physical device to  
2350 which the job is assigned.  
2351

2352 numberOfDocuments(33), Integer32\_ (-2..2147483647)  
2353 INTEGER: The number of documents in this job.  
2354

2355 The agent SHOULD return this attribute if the job has more  
2356 than one document.  
2357



```

2399     documentFormat(38),                               PrtInterpreterLangFamilyTC
2400                                             AND/OR
2401                                             OCTET STRING(SIZE(0..63))
2402     INTEGER: MULTI-ROW: The interpreter language family
2403     corresponding to the Printer MIB[print-mib]
2404     prtInterpreterLangFamily object, that this job
2405     requires/uses. A document or a job MAY use more than one
2406     PDL or control language.
2407
2408     AND/OR
2409
2410     OCTETS: MULTI-ROW: The document format registered as a
2411     media type[iana-media-types], i.e., the name of the MIME
2412     content-type/subtype. Examples: 'application/postscript',
2413     'application/vnd.hp-PCL', 'application/pdf', 'text/plain'
2414     (US-ASCII SHALL be assumed), 'text/plain; charset=iso-8859-
2415     1', and 'application/octet-stream'. The IPP 'document-
2416     format' job attribute uses these same values with the same
2417     semantics. See the IPP [ipp-model] 'mimeMediaType'
2418     attribute syntax and the document-format attribute for
2419     further examples and explanation.
2420
2421
2422     ++++++
2423     + Job Parameter attributes
2424     +
2425     + The following attributes represent input parameters
2426     + supplied by the submitting client in the job submission
2427     + protocol.
2428     ++++++
2429
2430     jobPriority(50),                                     Integer32 (-2..100)
2431     INTEGER: The priority for scheduling the job. It is used
2432     by servers and devices that employ a priority-based
2433     scheduling algorithm.
2434
2435     A higher value specifies a higher priority. The value 1 is
2436     defined to indicate the lowest possible priority (a job
2437     which a priority-based scheduling algorithm SHALL pass over
2438     in favor of higher priority jobs). The value 100 is
2439     defined to indicate the highest possible priority.
2440     Priority is expected to be evenly or 'normally' distributed
2441     across this range. The mapping of vendor-defined priority
2442     over this range is implementation-specific. -2 indicates
2443     unknown.
2444

```



2445           jobProcessAfterDateAndTime(51),     DateAndTime (SNMPv2-TC)  
2446           OCTETS: The calendar date and time of day after which the  
2447           job SHALL become a candidate to be scheduled for  
2448           processing. If the value of this attribute is in the  
2449           future, the server SHALL set the value of the job's  
2450           jmJobState object to pendingHeld and add the  
2451           jobProcessAfterSpecified bit value to the job's  
2452           jmJobStateReasons1 object. When the specified date and  
2453           time arrives, the server SHALL remove the  
2454           jobProcessAfterSpecified bit value from the job's  
2455           jmJobStateReasons1 object and, if no other reasons remain,  
2456           SHALL change the job's jmJobState object to pending.  
2457  
2458           jobHold(52),                            JmBooleanTC  
2459           INTEGER: If the value is 'true(4)', a client has  
2460           explicitly specified that the job is to be held until  
2461           explicitly released. Until the job is explicitly released  
2462           by a client, the job SHALL be in the pendingHeld state with  
2463           the jobHoldSpecified value in the jmJobStateReasons1  
2464           attribute.  
2465  
2466           jobHoldUntil(53),                      JmJobStringTC\_(SIZE(0..63))  
2467           OCTETS: The named time period during which the job SHALL  
2468           become a candidate for processing, such as 'evening',  
2469           'night', 'weekend', 'second-shift', 'third-shift', etc., as  
2470           defined by the system administrator. See IPP [ipp-model]  
2471           for the standard keyword values. Until that time period  
2472           arrives, the job SHALL be in the pendingHeld state with the  
2473           jobHoldUntilSpecified value in the jmJobStateReasons1  
2474           object. The value 'no-hold' SHALL indicate explicitly that  
2475           no time period has been specified; the absence of this  
2476           attribute SHALL indicate implicitly that no time period has  
2477           been specified.  
2478  
2479           outputBin(54),                          Integer32\_(0..2147483647)  
2480    AND/OR  
2481    JmJobStringTC\_(SIZE(0..63))  
2482           INTEGER: MULTI-ROW: The output subunit index in the  
2483           Printer MIB[print-mib]  
2484  
2485           AND/OR  
2486  
2487           OCTETS: MULTI-ROW: the name or number (represented as  
2488           ASCII digits) of the output bin to which all or part of the  
2489           job is placed in.  
2490

```
2491 sides(55), Integer32_(-2..2)
2492     INTEGER: MULTI-ROW: The number of sides, '1' or '2', that
2493     any document in this job requires/used.
2494
2495 finishing(56), JmFinishingTC
2496     INTEGER: MULTI-ROW: Type of finishing that any document
2497     in this job requires/used.
2498
2499
2500 ++++++
2501 + Image Quality attributes (requested and consumed)
2502 +
2503 + For devices that can vary the image quality.
2504 ++++++
2505
2506 printQualityRequested(70), JmPrintQualityTC
2507     INTEGER: MULTI-ROW: The print quality selection requested
2508     for a document in the job for printers that allow quality
2509     differentiation.
2510
2511 printQualityUsed(71), JmPrintQualityTC
2512     INTEGER: MULTI-ROW: The print quality selection actually
2513     used by a document in the job for printers that allow
2514     quality differentiation.
2515
2516 printerResolutionRequested(72), JmPrinterResolutionTC
2517     OCTETS: MULTI-ROW: The printer resolution requested for a
2518     document in the job for printers that support resolution
2519     selection.
2520
2521 printerResolutionUsed(73), JmPrinterResolutionTC
2522     OCTETS: MULTI-ROW: The printer resolution actually used
2523     by a document in the job for printers that support
2524     resolution selection.
2525
2526 tonerEcomonyRequested(74), JmTonerEcomonyTC
2527     INTEGER: MULTI-ROW: The toner economy selection requested
2528     for documents in the job for printers that allow toner
2529     economy differentiation.
2530
2531 tonerEcomonyUsed(75), JmTonerEcomonyTC
2532     INTEGER: MULTI-ROW: The toner economy selection actually
2533     used by documents in the job for printers that allow toner
2534     economy differentiation.
2535
2536 tonerDensityRequested(76), Integer32_(-2..100)
2537     INTEGER: MULTI-ROW: The toner density requested for a
2538     document in this job for devices that can vary toner
2539     density levels. Level 1 is the lowest density and level
2540     100 is the highest density level. Devices with a smaller
2541     range, SHALL map the 1-100 range evenly onto the
2542     implemented range.
```

2543  
2544 tonerDensityUsed(77), Integer32\_(-2..100)  
2545 INTEGER: MULTI-ROW: The toner density used by documents  
2546 in this job for devices that can vary toner density levels.  
2547 Level 1 is the lowest density and level 100 is the highest  
2548 density level. Devices with a smaller range, SHALL map the  
2549 1-100 range evenly onto the implemented range.  
2550  
2551  
2552 ++++++  
2553 + Job Progress attributes (requested and consumed)  
2554 +  
2555 + Pairs of these attributes can be used by monitoring  
2556 + applications to show an indication of relative progress  
2557 + to users. See section 3.4, entitled  
2558 + 'Monitoring Job Progress'.  
2559 ++++++  
2560  
2561 jobCopiesRequested(90), Integer32\_(-2..2147483647)  
2562 INTEGER: The number of copies of the entire job that are  
2563 to be produced.  
2564  
2565 jobCopiesCompleted(91), Integer32\_(-2..2147483647)  
2566 INTEGER: The number of copies of the entire job that have  
2567 been completed so far.  
2568  
2569 documentCopiesRequested(92), Integer32\_(-2..2147483647)  
2570 INTEGER: The total count of the number of document copies  
2571 requested for the job as a whole. If there are documents  
2572 A, B, and C, and document B is specified to produce 4  
2573 copies, the number of document copies requested is 6 for  
2574 the job.  
2575  
2576 This attribute SHALL be used only when a job has multiple  
2577 documents. The jobCopiesRequested attribute SHALL be used  
2578 when the job has only one document.  
2579  
2580 documentCopiesCompleted(93), Integer32\_(-2..2147483647)  
2581 INTEGER: The total count of the number of document copies  
2582 completed so far for the job as a whole. If there are  
2583 documents A, B, and C, and document B is specified to  
2584 produce 4 copies, the number of document copies starts a 0  
2585 and runs up to 6 for the job as the job processes.  
2586  
2587 This attribute SHALL be used only when a job has multiple  
2588 documents. The jobCopiesCompleted attribute SHALL be used  
2589 when the job has only one document.  
2590

2591           jobKOctetsTransferred(94),           Integer32\_(-2..2147483647)  
2592            INTEGER:  The number of K (1024) octets transferred to the  
2593            server or device to which the agent is providing access.  
2594            This count is independent of the number of copies of the  
2595            job or documents that will be produced, but it is only a  
2596            measure of the number of bytes transferred to the server or  
2597            device.  
2598  
2599            The agent SHALL round the actual number of octets  
2600            transferred up to the next higher K.  Thus 0 octets SHALL  
2601            be represented as '0', 1-1024 octets SHALL BE represented  
2602            as '1', 1025-2048 SHALL be '2', etc.  When the job  
2603            completes, the values of the jmJobKOctetsPerCopyRequested  
2604            object and the jobKOctetsTransferred attribute SHALL be  
2605            equal.  
2606  
2607            NOTE - The jobKOctetsTransferred can be used with the  
2608            jmJobKOctetsPerCopyRequested object in order to produce a  
2609            relative indication of the progress of the job for agents  
2610            that do not implement the jmJobKOctetsProcessed object.  
2611  
2612           sheetCompletedCopyNumber(95),           Integer32\_(-2..2147483647)  
2613            INTEGER:  The number of the copy being stacked for the  
2614            current document.  This number starts at 0, is set to 1  
2615            when the first sheet of the first copy for each document is  
2616            being stacked and is equal to n where n is the nth sheet  
2617            stacked in the current document copy.  See section 3.4 ,  
2618            entitled 'Monitoring Job Progress'.  
2619  
2620           sheetCompletedDocumentNumber(96), Integer32\_(-2..2147483647)  
2621            INTEGER:  The ordinal number of the document in the job  
2622            that is currently being stacked.  This number starts at 0,  
2623            increments to 1 when the first sheet of the first document  
2624            in the job is being stacked, and is equal to n where n is  
2625            the nth document in the job, starting with 1.  
2626  
2627            Implementations that only support one document jobs SHOULD  
2628            NOT implement this attribute.  
2629  
2630           jobCollationType(97),                    JmJobCollationTypeTC  
2631            INTEGER:  The type of job collation.  See also Section 3.4,  
2632            entitled 'Monitoring Job Progress'.  
2633

2634  
2635 ++++++  
2636 + Impression attributes  
2637 +  
2638 + See the definition of the terms 'impression', 'sheet',  
2639 + and 'page' in Section 2.  
2640 +  
2641 + See also jmJobImpressionsPerCopyRequested and  
2642 + jmJobImpressionsCompleted objects in the jmJobTable.  
2643 ++++++  
2644  
2645 impressionsSpooled(110), Integer32 (-2..2147483647)  
2646 INTEGER: The number of impressions spooled to the server  
2647 or device for the job so far.  
2648  
2649 impressionsSentToDevice(111), Integer32 (-2..2147483647)  
2650 INTEGER: The number of impressions sent to the device for  
2651 the job so far.  
2652  
2653 impressionsInterpreted(112), Integer32 (-2..2147483647)  
2654 INTEGER: The number of impressions interpreted for the job  
2655 so far.  
2656  
2657 impressionsCompletedCurrentCopy(113),  
2658 Integer32 (-2..2147483647)  
2659 INTEGER: The number of impressions completed by the device  
2660 for the current copy of the current document so far. For  
2661 printing, the impressions completed includes interpreting,  
2662 marking, and stacking the output. For other types of job  
2663 services, the number of impressions completed includes the  
2664 number of impressions processed.  
2665  
2666 This value SHALL be reset to 0 for each document in the job  
2667 and for each document copy.  
2668  
2669 fullColorImpressionsCompleted(114), Integer32 (-2..2147483647)  
2670 INTEGER: The number of full color impressions completed by  
2671 the device for this job so far. For printing, the  
2672 impressions completed includes interpreting, marking, and  
2673 stacking the output. For other types of job services, the  
2674 number of impressions completed includes the number of  
2675 impressions processed. Full color impressions are typically  
2676 defined as those requiring 3 or more colorants, but this  
2677 MAY vary by implementation. In any case, the value of this  
2678 attribute counts by 1 for each side that has full color,  
2679 not by the number of colors per side (and the other  
2680 impression counters are incremented, except  
2681 highlightColorImpressionsCompleted(115)).  
2682

2683 highlightColorImpressionsCompleted(115),  
2684 Integer32\_(-2..2147483647)  
2685 INTEGER: The number of highlight color impressions  
2686 completed by the device for this job so far. For printing,  
2687 the impressions completed includes interpreting, marking,  
2688 and stacking the output. For other types of job services,  
2689 the number of impressions completed includes the number of  
2690 impressions processed. Highlight color impressions are  
2691 typically defined as those requiring black plus one other  
2692 colorant, but this MAY vary by implementation. In any  
2693 case, the value of this attribute counts by 1 for each side  
2694 that has highlight color (and the other impression counters  
2695 are incremented, except  
2696 fullColorImpressionsCompleted(114)).  
2697  
2698  
2699 ++++++  
2700 + Page attributes  
2701 +  
2702 + See the definition of 'impression', 'sheet', and 'page'  
2703 + in Section 2.  
2704 ++++++

2705  
2706 pagesRequested(130), Integer32\_(-2..2147483647)  
2707 INTEGER: The number of logical pages requested by the job  
2708 to be processed.  
2709

2710 pagesCompleted(131), Integer32\_(-2..2147483647)  
2711 INTEGER: The number of logical pages completed for this  
2712 job so far.  
2713  
2714 For implementations where multiple copies are produced by  
2715 the interpreter with only a single pass over the data, the  
2716 final value SHALL be equal to the value of the  
2717 pagesRequested object. For implementations where multiple  
2718 copies are produced by the interpreter by processing the  
2719 data for each copy, the final value SHALL be a multiple of  
2720 the value of the pagesRequested object.  
2721

2722 NOTE - See the impressionsCompletedCurrentCopy and  
2723 pagesCompletedCurrentCopy attributes for attributes that  
2724 are reset on each document copy.  
2725

2726 NOTE - The pagesCompleted object can be used with the  
2727 pagesRequested object to provide an indication of the  
2728 relative progress of the job, provided that the  
2729 multiplicative factor is taken into account for some  
2730 implementations of multiple copies.  
2731

2732 pagesCompletedCurrentCopy(132), Integer32 (-2..2147483647)  
2733 INTEGER: The number of logical pages completed for the  
2734 current copy of the document so far. This value SHALL be  
2735 reset to 0 for each document in the job and for each  
2736 document copy.  
2737  
2738  
2739 ++++++  
2740 + Sheet attributes  
2741 +  
2742 + See the definition of 'impression', 'sheet', and 'page'  
2743 + in Section 2.  
2744 ++++++

2745  
2746 sheetsRequested(150), Integer32 (-2..2147483647)  
2747 INTEGER: The total number of medium sheets requested to be  
2748 produced for this job.  
2749  
2750 Unlike the jmJobKOctetsPerCopyRequested and  
2751 jmJobImpressionsPerCopyRequested attributes, the  
2752 sheetsRequested(150) attribute SHALL include the  
2753 multiplicative factor contributed by the number of copies  
2754 and so is the total number of sheets to be produced by the  
2755 job, as opposed to the size of the document(s) submitted.  
2756

2757 sheetsCompleted(151), Integer32 (-2..2147483647)  
2758 INTEGER: The total number of medium sheets that have  
2759 completed marking and stacking for the entire job so far  
2760 whether those sheets have been processed on one side or on  
2761 both.  
2762

2763 sheetsCompletedCurrentCopy(152), Integer32 (-2..2147483647)  
2764 INTEGER: The number of medium sheets that have completed  
2765 marking and stacking for the current copy of a document in  
2766 the job so far whether those sheets have been processed on  
2767 one side or on both.  
2768  
2769 The value of this attribute SHALL be 0 before the job  
2770 starts processing and SHALL be reset to 1 after the first  
2771 sheet of each document and document copy in the job is  
2772 processed and stacked.  
2773  
2774



```

2775      ++++++
2776      + Resources attributes (requested and consumed)
2777      +
2778      + Pairs of these attributes can be used by monitoring
2779      + applications to show an indication of relative usage to
2780      + users.
2781      ++++++
2782
2783      mediumRequested(170),                JmMediumTypeTC
2784                                          AND/OR
2785                                          JmJobStringTC_(SIZE(0..63))
2786      INTEGER:  MULTI-ROW:  The type
2787      AND/OR
2788      OCTETS:   MULTI-ROW:  the name of the medium that is
2789      required by the job.
2790
2791      NOTE - The name (JmJobStringTC) values correspond to the
2792      prtInputMediaName object in the Printer MIB [print-mib] and
2793      the values of the IPP 'media' attribute.
2794
2795      mediumConsumed(171),                Integer32_(-2..2147483647)
2796                                          AND
2797                                          JmJobStringTC_(SIZE(0..63))
2798      INTEGER:  MULTI-ROW:  The number of sheets
2799      AND
2800      OCTETS:   MULTI-ROW:  the name of the medium that has been
2801      consumed so far whether those sheets have been processed on
2802      one side or on both.
2803
2804      This attribute SHALL have both Integer32 and OCTET STRING
2805      (represented as JmJobStringTC) values.
2806
2807      NOTE - The name (JmJobStringTC) values correspond to the
2808      name values of the prtInputMediaName object in the Printer
2809      MIB [print-mib].
2810
2811      colorantRequested(172),              Integer32_(-2..2147483647)
2812                                          AND/OR
2813                                          JmJobStringTC_(SIZE(0..63))
2814      INTEGER:  MULTI-ROW:  The index (prtMarkerColorantIndex) in
2815      the Printer MIB[print-mib]
2816      AND/OR
2817      OCTETS:   MULTI-ROW:  the name of the colorant requested.
2818
2819      NOTE - The name (JmJobStringTC) values correspond to the
2820      name values of the prtMarkerColorantValue object in the
2821      Printer MIB.  Examples are: red, blue.

```

```

2822         colorantConsumed(173),           Integer32 (-2..2147483647)
2823                                         AND/OR
2824                                         JmJobStringTC (SIZE(0..63))
2825         INTEGER: MULTI-ROW: The index (prtMarkerColorantIndex) in
2826         the Printer MIB[print-mib]
2827         AND/OR
2828         OCTETS: MULTI-ROW: the name of the colorant consumed.
2829
2830         NOTE - The name (JmJobStringTC) values correspond to the
2831         name values of the prtMarkerColorantValue object in the
2832         Printer MIB. Examples are: red, blue
2833
2834
2835         ++++++
2836         + Time attributes (set by server or device)
2837         +
2838         + This section of attributes are ones that are set by the
2839         + server or device that accepts jobs. Two forms of time are
2840         + provided. Each form is represented in a separate attribute.
2841         + See section 3.1.2 and section 3.1.3 for the
2842         + conformance requirements for time attribute for agents and
2843         + monitoring applications, respectively. The two forms are:
2844         +
2845         + 'DateAndTime' is an 8 or 11 octet binary encoded year,
2846         + month, day, hour, minute, second, deci-second with
2847         + optional offset from UTC. See SNMPv2-TC [SMIV2-TC].
2848         +
2849         + NOTE: 'DateAndTime' is not printable characters; it is
2850         + binary.
2851         +
2852         + 'JmTimeStampTC' is the time of day measured in the number of
2853         + seconds since the system was booted.
2854         ++++++
2855
2856         jobSubmissionToServerTime(190),     JmTimeStampTC
2857                                         AND/OR
2858                                         DateAndTime
2859         INTEGER: Configuration 3 only: The time
2860         AND/OR
2861         OCTETS: the date and time that the job was submitted to
2862         the server (as distinguished from the device which uses
2863         jobSubmissionTime).
2864
2865         jobSubmissionTime(191),             JmTimeStampTC
2866                                         AND/OR
2867                                         DateAndTime
2868         INTEGER: Configurations 1, 2, and 3: The time
2869         AND/OR
2870         OCTETS: the date and time that the job was submitted to
2871         the server or device to which the agent is providing
2872         access.
2873

```

2874           jobStartedBeingHeldTime(192),           JmTimeStampTC  
2875    AND/OR  
2876    DateAndTime  
2877            INTEGER:   The time  
2878            AND/OR  
2879            OCTETS:   the date and time that the job last entered the  
2880            pendingHeld state.  If the job has never entered the  
2881            pendingHeld state, then the value SHALL be '0' or the  
2882            attribute SHALL not be present in the table.  
2883  
2884           jobStartedProcessingTime(193),           JmTimeStampTC  
2885    AND/OR  
2886    DateAndTime  
2887            INTEGER:   The time  
2888            AND/OR  
2889            OCTETS:   the date and time that the job started processing.  
2890  
2891           jobCompletionTime(194),                JmTimeStampTC  
2892    AND/OR  
2893    DateAndTime  
2894            INTEGER:   The time  
2895            AND/OR  
2896            OCTETS:   the date and time that the job entered the  
2897            completed, canceled, or aborted state.  
2898  
2899           jobProcessingCPUtime(195)                Integer32 (-2..2147483647)  
2900    UNITS    'seconds'  
2901            INTEGER:   The amount of CPU time in seconds that the job  
2902            has been in the processing state.  If the job enters the  
2903            processingStopped state, that elapsed time SHALL not be  
2904            included.  In other words, the jobProcessingCPUtime value  
2905            SHOULD be relatively repeatable when the same job is  
2906            processed again on the same device."  
2907  
2908   REFERENCE  
2909        "See Section 3.2 entitled 'The Attribute Mechanism' for a  
2910        description of this textual-convention and its use in the  
2911        jmAttributeTable.  
2912  
2913        This is a type 2 enumeration.  See Section 3.7.1.2."  
2914   SYNTAX        INTEGER {  
2915        other(1),  
2916  
2917        -- Job State attributes:  
2918        jobStateReasons2(3),  
2919        jobStateReasons3(4),  
2920        jobStateReasons4(5),  
2921        processingMessage(6),  
2922        processingMessageNaturalLangTag(7),  
2923        jobCodedCharSet(8),  
2924        jobNaturalLanguageTag(9),  
2925

```
2926     -- Job Identification attributes:
2927     jobURI(20),
2928     jobAccountName(21),
2929     serverAssignedJobName(22),
2930     jobName(23),
2931     jobServiceTypes(24),
2932     jobSourceChannelIndex(25),
2933     jobSourcePlatformType(26),
2934     submittingServerName(27),
2935     submittingApplicationName(28),
2936     jobOriginatingHost(29),
2937     deviceNameRequested(30),
2938     queueNameRequested(31),
2939     physicalDevice(32),
2940     numberOfDocuments(33),
2941     fileName(34),
2942     documentName(35),
2943     jobComment(36),
2944     documentFormatIndex(37),
2945     documentFormat(38),
2946
2947     -- Job Parameter attributes:
2948     jobPriority(50),
2949     jobProcessAfterDateAndTime(51),
2950     jobHold(52),
2951     jobHoldUntil(53),
2952     outputBin(54),
2953     sides(55),
2954     finishing(56),
2955
2956     -- Image Quality attributes:
2957     printQualityRequested(70),
2958     printQualityUsed(71),
2959     printerResolutionRequested(72),
2960     printerResolutionUsed(73),
2961     tonerEcomonyRequested(74),
2962     tonerEcomonyUsed(75),
2963     tonerDensityRequested(76),
2964     tonerDensityUsed(77),
2965
2966     -- Job Progress attributes:
2967     jobCopiesRequested(90),
2968     jobCopiesCompleted(91),
2969     documentCopiesRequested(92),
2970     documentCopiesCompleted(93),
2971     jobKOctetsTransferred(94),
2972     sheetCompletedCopyNumber(95),
2973     sheetCompletedDocumentNumber(96),
2974     jobCollationType(97),
2975
```

```
2976     -- Impression attributes:
2977     impressionsSpooled(110),
2978     impressionsSentToDevice(111),
2979     impressionsInterpreted(112),
2980     impressionsCompletedCurrentCopy(113),
2981     fullColorImpressionsCompleted(114),
2982     highlightColorImpressionsCompleted(115),
2983
2984     -- Page attributes:
2985     pagesRequested(130),
2986     pagesCompleted(131),
2987     pagesCompletedCurrentCopy(132),
2988
2989     -- Sheet attributes:
2990     sheetsRequested(150),
2991     sheetsCompleted(151),
2992     sheetsCompletedCurrentCopy(152),
2993
2994     -- Resource attributes:
2995     mediumRequested(170),
2996     mediumConsumed(171),
2997     colorantRequested(172),
2998     colorantConsumed(173),
2999
3000     -- Time attributes:
3001     jobSubmissionToServerTime(190),
3002     jobSubmissionTime(191),
3003     jobStartedBeingHeldTime(192),
3004     jobStartedProcessingTime(193),
3005     jobCompletionTime(194),
3006     jobProcessingCPUTime(195)
3007 }
3008
3009
3010
3011
```

3012 JmJobServiceTypesTC ::= TEXTUAL-CONVENTION  
3013     STATUS       current  
3014     DESCRIPTION  
3015         "Specifies the type(s) of service to which the job has been  
3016         submitted (print, fax, scan, etc.). The service type is  
3017         represented as an enum that is bit encoded with each job  
3018         service type so that more general and arbitrary services can be  
3019         created, such as services with more than one destination type,  
3020         or ones with only a source or only a destination. For example,  
3021         a job service might scan, faxOut, and print a single job. In  
3022         this case, three bits would be set in the jobServiceTypes  
3023         attribute, corresponding to the hexadecimal values: 0x8 + 0x20  
3024         + 0x4, respectively, yielding: 0x2C.  
3025  
3026         Whether this attribute is set from a job attribute supplied by  
3027         the job submission client or is set by the recipient job  
3028         submission server or device depends on the job submission  
3029         protocol. With either implementation, the agent SHALL return a  
3030         non-zero value for this attribute indicating the type of the  
3031         job.  
3032  
3033         One of the purposes of this attribute is to permit a requester  
3034         to filter out jobs that are not of interest. For example, a  
3035         printer operator MAY only be interested in jobs that include  
3036         printing. That is why the attribute is in the job  
3037         identification category.  
3038  
3039         The following service component types are defined (in  
3040         hexadecimal) and are assigned a separate bit value for use with  
3041         the jobServiceTypes attribute:  
3042  
3043         other                                 0x1  
3044             The job contains some instructions that are not one of the  
3045             identified types.  
3046  
3047         unknown                             0x2  
3048             The job contains some instructions whose type is unknown to  
3049             the agent.  
3050  
3051         print                               0x4  
3052             The job contains some instructions that specify printing  
3053  
3054         scan                                0x8  
3055             The job contains some instructions that specify scanning  
3056  
3057         faxIn                               0x10  
3058             The job contains some instructions that specify receive fax  
3059  
3060         faxOut                             0x20  
3061             The job contains some instructions that specify sending fax  
3062





3115  
3116 other 0x1  
3117 The job state reason is not one of the standardized or  
3118 registered reasons.  
3119  
3120 unknown 0x2  
3121 The job state reason is not known to the agent or is  
3122 indeterminent.  
3123  
3124 jobIncoming 0x4  
3125 The job has been accepted by the server or device, but the  
3126 server or device is expecting (1) additional operations  
3127 from the client to finish creating the job and/or (2) is  
3128 accessing/accepting document data.  
3129  
3130 submissionInterrupted 0x8  
3131 The job was not completely submitted for some unforeseen  
3132 reason, such as: (1) the server has crashed before the job  
3133 was closed by the client, (2) the server or the document  
3134 transfer method has crashed in some non-recoverable way  
3135 before the document data was entirely transferred to the  
3136 server, (3) the client crashed or failed to close the job  
3137 before the time-out period.  
3138  
3139 jobOutgoing 0x10  
3140 Configuration 2 only: The server is transmitting the job  
3141 to the device.  
3142  
3143 jobHoldSpecified 0x20  
3144 The value of the job's jobHold(52) attribute is TRUE. The  
3145 job SHALL NOT be a candidate for processing until this  
3146 reason is removed and there are no other reasons to hold  
3147 the job.  
3148  
3149 jobHoldUntilSpecified 0x40  
3150 The value of the job's jobHoldUntil(53) attribute specifies  
3151 a time period that is still in the future. The job SHALL  
3152 NOT be a candidate for processing until this reason is  
3153 removed and there are no other reasons to hold the job.  
3154  
3155 jobProcessAfterSpecified 0x80  
3156 The value of the job's jobProcessAfterDateAndTime(51)  
3157 attribute specifies a time that is still in the future.  
3158 The job SHALL NOT be a candidate for processing until this  
3159 reason is removed and there are no other reasons to hold  
3160 the job.  
3161

3162 resourcesAreNotReady 0x100  
3163 At least one of the resources needed by the job, such as  
3164 media, fonts, resource objects, etc., is not ready on any  
3165 of the physical devices for which the job is a candidate.  
3166 This condition MAY be detected when the job is accepted, or  
3167 subsequently while the job is pending or processing,  
3168 depending on implementation.  
3169  
3170 deviceStoppedPartly 0x200  
3171 One or more, but not all, of the devices to which the job  
3172 is assigned are stopped. If all of the devices are stopped  
3173 (or the only device is stopped), the deviceStopped reason  
3174 SHALL be used.  
3175  
3176 deviceStopped 0x400  
3177 The device(s) to which the job is assigned is (are all)  
3178 stopped.  
3179  
3180 jobInterpreting 0x800  
3181 The device to which the job is assigned is interpreting the  
3182 document data.  
3183  
3184 jobPrinting 0x1000  
3185 The output device to which the job is assigned is marking  
3186 media. This ~~attribute-value~~ is useful for servers and  
3187 output devices which spend a great deal of time processing  
3188 (1) when no marking is happening and then want to show that  
3189 marking is now happening or (2) when the job is in the  
3190 process of being canceled or aborted while the job remains  
3191 in the processing state, but the marking has not yet  
3192 stopped so that impression or sheet counts are still  
3193 increasing for the job.  
3194  
3195 jobCanceledByUser 0x2000  
3196 The job was canceled by the owner of the job, i.e., by a  
3197 user whose name is the same as the value of the job's  
3198 jmJobOwner object, or by some other authorized end-user,  
3199 such as a member of the job owner's security group.  
3200  
3201 jobCanceledByOperator 0x4000  
3202 The job was canceled by the operator, i.e., by a user who  
3203 has been authenticated as having operator privileges  
3204 (whether local or remote).  
3205  
3206 jobCanceledAtDevice 0x8000  
3207 The job was canceled by an unidentified local user, i.e., a  
3208 user at a console at the device.  
3209

3210 abortedBySystem 0x10000  
3211 The job (1) is in the process of being aborted, (2) has  
3212 been aborted by the system and placed in the 'aborted'  
3213 state, or (3) has been aborted by the system and placed in  
3214 the 'pendingHeld' state, so that a user or operator can  
3215 manually try the job again.  
3216

3217 processingToStopPoint 0x20000  
3218 The requester has issued an operation to cancel or  
3219 interrupt the job or the server/device has aborted the job,  
3220 but the server/device is still performing some actions on  
3221 the job until a specified stop point occurs or job  
3222 termination/cleanup is completed.  
3223

3224 This reason is recommended to be used in conjunction with  
3225 the processing job state to indicate that the server/device  
3226 is still performing some actions on the job while the job  
3227 remains in the processing state. After all the job's  
3228 resources consumed counters have stopped incrementing, the  
3229 server/device moves the job from the processing state to  
3230 the canceled or aborted job states.  
3231

3232 serviceOffLine 0x40000  
3233 The service or document transform is off-line and accepting  
3234 no jobs. All pending jobs are put into the pendingHeld  
3235 state. This situation could be true if the service's or  
3236 document transform's input is impaired or broken.  
3237

3238 jobCompletedSuccessfully 0x80000  
3239 The job completed successfully.  
3240

3241 jobCompletedWithWarnings 0x100000  
3242 The job completed with warnings.  
3243

3244 jobCompletedWithErrors 0x200000  
3245 The job completed with errors (and possibly warnings too).  
3246  
3247

3248 The following additional job state reasons have been added to  
3249 represent job states that are in ISO DPA[iso-dpa] and other job  
3250 submission protocols:  
3251

3252 jobPaused 0x400000  
3253 The job has been indefinitely suspended by a client issuing  
3254 an operation to suspend the job so that other jobs may  
3255 proceed using the same devices. The client MAY issue an  
3256 operation to resume the paused job at any time, in which  
3257 case the agent SHALL remove the jobPaused values from the  
3258 job's jmJobStateReasons1 object and the job is eventually  
3259 resumed at or near the point where the job was paused.  
3260

```

3261     jobInterrupted                0x800000
3262     The job has been interrupted while processing by a client
3263     issuing an operation that specifies another job to be run
3264     instead of the current job.  The server or device will
3265     automatically resume the interrupted job when the
3266     interrupting job completes.
3267
3268     jobRetained                      0x1000000
3269     The job is being retained by the server or device with all
3270     of the job's document data (and submitted resources, such
3271     as fonts, logos, and forms, if any).  Thus a client could
3272     issue an operation to the server or device to either (1)
3273     re-do the job (or a copy of the job) on the same server or
3274     device or (2) resubmit the job to another server or device.
3275     When a client could no longer re-do/resubmit the job, such
3276     as after the document data has been discarded, the agent
3277     SHALL remove the jobRetained value from the
3278     jmJobStateReasons1 object."
3279 REFERENCE
3280     "These bit definitions are the equivalent of a type 2 enum
3281     except that combinations of bits may be used together.  See
3282     section 3.7.1.2.  The remaining bits are reserved for future
3283     standardization and/or registration."
3284 SYNTAX      INTEGER_(0..2147483647)  -- 31 bits, all but sign bit
3285
3286
3287
3288 JmJobStateReasons2TC ::= TEXTUAL-CONVENTION
3289     STATUS      current
3290     DESCRIPTION
3291     "This textual-convention is used with the jobStateReasons2
3292     attribute to provides additional information regarding the
3293     jmJobState object.  See the description under
3294     JmJobStateReasons1TC for additional information that applies to
3295     all reasons.
3296
3297     The following standard values are defined (in hexadecimal) as
3298     powers of two, since multiple values may be used at the same
3299     time:
3300
3301     cascaded                0x1
3302     An outbound gateway has transmitted all of the job's job
3303     and document attributes and data to another spooling
3304     system.
3305
3306     deletedByAdministrator  0x2
3307     The administrator has deleted the job.
3308
3309     discardTimeArrived     0x4
3310     The job has been deleted due to the fact that the time
3311     specified by the job's job-discard-time attribute has
3312     arrived.

```

3313  
3314       postProcessingFailed                   0x8  
3315       The post-processing agent failed while trying to log  
3316       accounting attributes for the job; therefore the job has  
3317       been placed into the completed state with the jobRetained  
3318       jmJobStateReasons1 object value for a system-defined period  
3319       of time, so the administrator can examine it, resubmit it,  
3320       etc.  
3321  
3322       jobTransforming                        0x10  
3323       The server/device is interpreting document data and  
3324       producing another electronic representation.  
3325  
3326       maxJobFaultCountExceeded            0x20  
3327       The job has faulted several times and has exceeded the  
3328       administratively defined fault count limit.  
3329  
3330       devicesNeedAttentionTimeOut         0x40  
3331       One or more document transforms that the job is using needs  
3332       human intervention in order for the job to make progress,  
3333       but the human intervention did not occur within the site-  
3334       settable time-out value.  
3335  
3336       needsKeyOperatorTimeOut             0x80  
3337       One or more devices or document transforms that the job is  
3338       using need a specially trained operator (who may need a key  
3339       to unlock the device and gain access) in order for the job  
3340       to make progress, but the key operator intervention did not  
3341       occur within the site-settable time-out value.  
3342  
3343       jobStartWaitTimeOut                 0x100  
3344       The server/device has stopped the job at the beginning of  
3345       processing to await human action, such as installing a  
3346       special cartridge or special non-standard media, but the  
3347       job was not resumed within the site-settable time-out value  
3348       and the server/device has transitioned the job to the  
3349       pendingHeld state.  
3350  
3351       jobEndWaitTimeOut                    0x200  
3352       The server/device has stopped the job at the end of  
3353       processing to await human action, such as removing a  
3354       special cartridge or restoring standard media, but the job  
3355       was not resumed within the site-settable time-out value and  
3356       the server/device has transitioned the job to the completed  
3357       state.  
3358  
3359       jobPasswordWaitTimeOut              0x400  
3360       The server/device has stopped the job at the beginning of  
3361       processing to await input of the job's password, but the  
3362       password was not received within the site-settable time-out  
3363       value.  
3364

3365 deviceTimedOut 0x800  
3366 A device that the job was using has not responded in a  
3367 period specified by the device's site-settable attribute.  
3368  
3369 connectingToDeviceTimeOut 0x1000  
3370 The server is attempting to connect to one or more devices  
3371 which may be dial-up, polled, or queued, and so may be busy  
3372 with traffic from other systems, but server was unable to  
3373 connect to the device within the site-settable time-out  
3374 value.  
3375  
3376 transferring 0x2000  
3377 The job is being transferred to a down stream server or  
3378 downstream device.  
3379  
3380 queuedInDevice 0x4000  
3381 The server/device has queued the job in a down stream  
3382 server or downstream device.  
3383  
3384 jobQueued 0x8000  
3385 The server/device has queued the document data.  
3386  
3387 jobCleanup 0x10000  
3388 The server/device is performing cleanup activity as part of  
3389 ending normal processing.  
3390  
3391 jobPasswordWait 0x20000  
3392 The server/device has selected the job to be next to  
3393 process, but instead of assigning resources and starting  
3394 the job processing, the server/device has transitioned the  
3395 job to the pendingHeld state to await entry of a password  
3396 (and dispatched another job, if there is one).  
3397  
3398 validating 0x40000  
3399 The server/device is validating the job *after* accepting the  
3400 job.  
3401  
3402 queueHeld 0x80000  
3403 The operator has held the entire job set or queue.  
3404  
3405 jobProofWait 0x100000  
3406 The job has produced a single proof copy and is in the  
3407 pendingHeld state waiting for the requester to issue an  
3408 operation to release the job to print normally, obeying any  
3409 job and document copy attributes that were originally  
3410 submitted.  
3411  
3412 heldForDiagnostics 0x200000  
3413 The system is running intrusive diagnostics, so that all  
3414 jobs are being held.

3415 noSpaceOnServer 0x800000  
3416 There is no room on the server to store all of the job.  
3417  
3418 pinRequired 0x1000000  
3419 The System Administrator settable device policy is (1) to  
3420 require PINs, and (2) to hold jobs that do not have a pin  
3421 supplied as an input parameter when the job was created.  
3422  
3423 exceededAccountLimit 0x2000000  
3424 The account for which this job is drawn has exceeded its  
3425 limit. This condition SHOULD be detected before the job is  
3426 scheduled so that the user does not wait until his/her job  
3427 is scheduled only to find that the account is overdrawn.  
3428 This condition MAY also occur while the job is processing  
3429 either as processing begins or part way through processing.  
3430  
3431 heldForRetry 0x4000000  
3432 The job encountered some errors that the server/device  
3433 could not recover from with its normal retry procedures,  
3434 but the error might not be encountered if the job is  
3435 processed again in the future. Example cases are phone  
3436 number busy or remote file system in-accessible. For such  
3437 a situation, the server/device SHALL transition the job  
3438 from the processing to the pendingHeld, rather than to the  
3439 aborted state.  
3440  
3441 The following values are from the X/Open PSIS draft standard:  
3442  
3443 canceledByShutdown 0x8000000  
3444 The job was canceled because the server or device was  
3445 shutdown before completing the job.  
3446  
3447 deviceUnavailable 0x10000000  
3448 This job was aborted by the system because the device is  
3449 currently unable to accept jobs.  
3450  
3451 wrongDevice 0x20000000  
3452 This job was aborted by the system because the device is  
3453 unable to handle this particular job; the spooler SHOULD  
3454 try another device or the user should submit the job to  
3455 another device.  
3456  
3457 badJob 0x40000000  
3458 This job was aborted by the system because this job has a  
3459 major problem, such as an ill-formed PDL; the spooler  
3460 SHOULD not even try another device. "  
3461 REFERENCE  
3462 "These bit definitions are the equivalent of a type 2 enum  
3463 except that combinations of them may be used together. See  
3464 section 3.7.1.2. See the description under  
3465 JmJobStateReasons1TC and the jobStateReasons2 attribute."  
3466 SYNTAX INTEGER\_(0..2147483647) -- 31 bits, all but sign bit |



3467  
3468 JmJobStateReasons3TC ::= TEXTUAL-CONVENTION  
3469     STATUS         current  
3470     DESCRIPTION  
3471         "This textual-convention is used with the jobStateReasons3  
3472         attribute to provides additional information regarding the  
3473         jmJobState object. See the description under  
3474         JmJobStateReasons1TC for additional information that applies to  
3475         all reasons.  
3476  
3477         The following standard values are defined (in hexadecimal) as  
3478         *powers of two*, since multiple values may be used at the same  
3479         time:  
3480  
3481         jobInterruptedByDeviceFailure         0x1  
3482             A device or the print system software that the job was  
3483             using has failed while the job was processing. The server  
3484             or device is keeping the job in the pendingHeld state until  
3485             an operator can determine what to do with the job."  
3486     REFERENCE  
3487         "These bit definitions are the equivalent of a type 2 enum  
3488         except that combinations of them may be used together. See  
3489         section 3.7.1.2. The remaining bits are reserved for future  
3490         standardization and/or registration. See the description under  
3491         JmJobStateReasons1TC and the jobStateReasons3 attribute."  
3492     SYNTAX         INTEGER\_(0..2147483647)     -- 31 bits, all but sign bit  
3493  
3494  
3495  
3496  
3497  
3498 JmJobStateReasons4TC ::= TEXTUAL-CONVENTION  
3499     STATUS         current  
3500     DESCRIPTION  
3501         "This textual-convention is used in the jobStateReasons4  
3502         attribute to provides additional information regarding the  
3503         jmJobState object. See the description under  
3504         JmJobStateReasons1TC for additional information that applies to  
3505         all reasons.  
3506  
3507         The following standard values are defined (in hexadecimal) as  
3508         *powers of two*, since multiple values may be used at the same  
3509         time:  
3510  
3511         none yet defined. These bits are reserved for future  
3512         standardization and/or registration."  
3513     REFERENCE  
3514         "These bit definitions are the equivalent of a type 2 enum  
3515         except that combinations of them may be used together. See  
3516         section 3.7.1.2. See the description under  
3517         JmJobStateReasons1TC and the jobStateReasons4 attribute."  
3518     SYNTAX         INTEGER\_(0..2147483647)     -- 31 bits, all but sign bit

```

3519
3520 jobmonMIBObjects OBJECT IDENTIFIER ::= { jobmonMIB 1 }
3521
3522 -- The General Group (MANDATORY)
3523
3524 -- The jmGeneralGroup consists entirely of the jmGeneralTable.
3525
3526 jmGeneral OBJECT IDENTIFIER ::= { jobmonMIBObjects 1 }
3527
3528 jmGeneralTable OBJECT-TYPE
3529     SYNTAX      SEQUENCE OF JmGeneralEntry
3530     MAX-ACCESS  not-accessible
3531     STATUS      current
3532     DESCRIPTION
3533         "The jmGeneralTable consists of information of a general nature
3534         that are per-job-set, but are not per-job. See Section 2
3535         entitled 'Terminology and Job Model' for the definition of a
3536         job set."
3537     REFERENCE
3538         "The MANDATORY-GROUP macro specifies that this group is
3539         MANDATORY."
3540     ::= { jmGeneral 1 }
3541
3542
3543 jmGeneralEntry OBJECT-TYPE
3544     SYNTAX      JmGeneralEntry
3545     MAX-ACCESS  not-accessible
3546     STATUS      current
3547     DESCRIPTION
3548         "Information about a job set (queue).
3549
3550         An entry SHALL exist in this table for each job set."
3551     INDEX      { jmGeneralJobSetIndex }
3552     ::= { jmGeneralTable 1 }
3553
3554
3555 JmGeneralEntry ::= SEQUENCE {
3556     jmGeneralJobSetIndex      Integer32 (1..32767),
3557     jmGeneralNumberOfActiveJobs Integer32 (0..2147483647),
3558     jmGeneralOldestActiveJobIndex Integer32 (0..2147483647),
3559     jmGeneralNewestActiveJobIndex Integer32 (0..2147483647),
3560     jmGeneralJobPersistence   Integer32 (15..2147483647),
3561     jmGeneralAttributePersistence Integer32 (15..2147483647),
3562     jmGeneralJobSetName      JmUTF8StringTC (SIZE(0..63))
3563 }
3564

```

```
3565 jmGeneralJobSetIndex OBJECT-TYPE
3566     SYNTAX      Integer32 (1..32767)
3567     MAX-ACCESS  not-accessible
3568     STATUS      current
3569     DESCRIPTION
3570         "A unique value for each job set in this MIB.  The jmJobTable
3571         and jmAttributeTable tables have this same index as their
3572         primary index.
3573
3574         The value(s) of the jmGeneralJobSetIndex SHALL be persistent
3575         across power cycles, so that clients that have retained
3576         jmGeneralJobSetIndex values will access the same job sets upon
3577         subsequent power-up.
3578
3579         An implementation that has only one job set, such as a printer
3580         with a single queue, SHALL hard code this object with the value
3581         1."
3582     REFERENCE
3583         "See Section 2 entitled 'Terminology and Job Model' for the
3584         definition of a job set.
3585         Corresponds to the first index in jmJobTable and
3586         jmAttributeTable."
3587     ::= { jmGeneralEntry 1 }
3588
3589
3590 jmGeneralNumberOfActiveJobs OBJECT-TYPE
3591     SYNTAX      Integer32 (0..2147483647)
3592     MAX-ACCESS  read-only
3593     STATUS      current
3594     DESCRIPTION
3595         "The current number of 'active' jobs in the jmJobIDTable,
3596         jmJobTable, and jmAttributeTable, i.e., the total number of
3597         jobs that are in the pending, processing, or processingStopped
3598         states.  See the JmJobStateTC textual-convention for the exact
3599         specification of the semantics of the job states."
3600     DEFVAL     { 0 }      -- no jobs
3601     ::= { jmGeneralEntry 2 }
3602
```

```
3603 jmGeneralOldestActiveJobIndex OBJECT-TYPE
3604     SYNTAX      Integer32 (0..2147483647)
3605     MAX-ACCESS  read-only
3606     STATUS      current
3607     DESCRIPTION
3608         "The jmJobIndex of the oldest job that is still in one of the
3609         'active' states (pending, processing, or processingStopped).
3610         In other words, the index of the 'active' job that has been in
3611         the job tables the longest.
3612
3613         If there are no active jobs, the agent SHALL set the value of
3614         this object to 0."
3615     REFERENCE
3616         "See Section 3.2 entitled 'The Job Tables and the Oldest Active
3617         and Newest Active Indexes' for a description of the usage of
3618         this object."
3619     DEFVAL      { 0 }      -- no active jobs
3620     ::= { jmGeneralEntry 3 }
3621
3622
3623
3624 jmGeneralNewestActiveJobIndex OBJECT-TYPE
3625     SYNTAX      Integer32 (0..2147483647)
3626     MAX-ACCESS  read-only
3627     STATUS      current
3628     DESCRIPTION
3629         "The jmJobIndex of the newest job that is in one of the
3630         'active' states (pending, processing, or processingStopped).
3631         In other words, the index of the 'active' job that has been
3632         most recently added to the job tables.
3633
3634         When all jobs become 'inactive', i.e., enter the pendingHeld,
3635         completed, canceled, or aborted states, the agent SHALL set the
3636         value of this object to 0."
3637     REFERENCE
3638         "See Section 3.2 entitled 'The Job Tables and the Oldest Active
3639         and Newest Active Indexes' for a description of the usage of
3640         this object."
3641     DEFVAL      { 0 }      -- no active jobs
3642     ::= { jmGeneralEntry 4 }
3643
```

```
3644 jmGeneralJobPersistence OBJECT-TYPE
3645     SYNTAX      Integer32 (15..2147483647)
3646     UNITS       "seconds"
3647     MAX-ACCESS  read-only
3648     STATUS      current
3649     DESCRIPTION
3650         "The minimum time in seconds for this instance of the Job Set
3651         that an entry SHALL remain in the jmJobIDTable and jmJobTable
3652         after processing has completed, i.e., the minimum time in
3653         seconds starting when the job enters the completed, canceled,
3654         or aborted state.
3655
3656         Configuring this object is implementation-dependent.
3657
3658         This value SHALL be equal to or greater than the value of
3659         jmGeneralAttributePersistence. This value SHOULD be at least
3660         60 which gives a monitoring application one minute in which to
3661         poll for job data."
3662     DEFVAL      { 60 }          -- one minute
3663     ::= { jmGeneralEntry 5 }
3664
3665
3666
3667 jmGeneralAttributePersistence OBJECT-TYPE
3668     SYNTAX      Integer32 (15..2147483647)
3669     UNITS       "seconds"
3670     MAX-ACCESS  read-only
3671     STATUS      current
3672     DESCRIPTION
3673         "The minimum time in seconds for this instance of the Job Set
3674         that an entry SHALL remain in the jmAttributeTable after
3675         processing has completed , i.e., the time in seconds starting
3676         when the job enters the completed, canceled, or aborted state.
3677
3678         Configuring this object is implementation-dependent.
3679
3680         This value SHOULD be at least 60 which gives a monitoring
3681         application one minute in which to poll for job data."
3682     DEFVAL      { 60 }          -- one minute
3683     ::= { jmGeneralEntry 6 }
3684
```

```
3685 jmGeneralJobSetName OBJECT-TYPE
3686     SYNTAX      JmUTF8StringTC_(SIZE(0..63))
3687     MAX-ACCESS  read-only
3688     STATUS      current
3689     DESCRIPTION
3690         "The human readable name of this job set assigned by the system
3691         administrator (by means outside of this MIB).  Typically, this
3692         name SHOULD be the name of the job queue.  If a server or
3693         device has only a single job set, this object can be the
3694         administratively assigned name of the server or device itself.
3695         This name does not need to be unique, though each job set in a
3696         single Job Monitoring MIB SHOULD have distinct names.
3697
3698         NOTE - If the job set corresponds to a single printer and the
3699         Printer MIB is implemented, this value SHOULD be the same as
3700         the prtGeneralPrinterName object in the draft Printer MIB
3701         [print-mib-draft].  If the job set corresponds to an IPP
3702         Printer, this value SHOULD be the same as the IPP 'printer-
3703         name' Printer attribute.
3704
3705         NOTE - The purpose of this object is to help the user of the
3706         job monitoring application distinguish between several job sets
3707         in implementations that support more than one job set."
3708     REFERENCE
3709         "See the OBJECT compliance macro for the minimum maximum length
3710         required for conformance."
3711     DEFVAL      { 'H } -- empty string
3712     ::= { jmGeneralEntry 7 }
3713
3714
3715
3716
3717
```

```

3718 -- The Job ID Group (MANDATORY)
3719
3720 -- The jmJobIDGroup consists entirely of the jmJobIDTable.
3721
3722 jmJobID OBJECT IDENTIFIER ::= { jobmonMIBObjects 2 }
3723
3724 jmJobIDTable OBJECT-TYPE
3725     SYNTAX      SEQUENCE OF JmJobIDEntry
3726     MAX-ACCESS  not-accessible
3727     STATUS      current
3728     DESCRIPTION
3729         "The jmJobIDTable provides a correspondence map (1) between the
3730         job submission ID that a client uses to refer to a job and (2)
3731         the jmGeneralJobSetIndex and jmJobIndex that the Job Monitoring
3732         MIB agent assigned to the job and that are used to access the
3733         job in all of the other tables in the MIB.  If a monitoring
3734         application already knows the jmGeneralJobSetIndex and the
3735         jmJobIndex of the job it is querying, that application NEED NOT
3736         use the jmJobIDTable."
3737     REFERENCE
3738         "The MANDATORY-GROUP macro specifies that this group is
3739         MANDATORY."
3740     ::= { jmJobID 1 }
3741
3742
3743
3744 jmJobIDEntry OBJECT-TYPE
3745     SYNTAX      JmJobIDEntry
3746     MAX-ACCESS  not-accessible
3747     STATUS      current
3748     DESCRIPTION
3749         "The map from (1) the jmJobSubmissionID to (2) the
3750         jmGeneralJobSetIndex and jmJobIndex.
3751
3752         An entry SHALL exist in this table for each job currently known
3753         to the agent for all job sets and job states.  There MAY be
3754         more than one jmJobIDEntry that maps to a single job.  This
3755         many to one mapping can occur when more than one network entity
3756         along the job submission path supplies a job submission ID.
3757         See Section 3.5.  However, each job SHALL appear once and in
3758         one and only one job set."
3759     INDEX { jmJobSubmissionID }
3760     ::= { jmJobIDTable 1 }
3761
3762 JmJobIDEntry ::= SEQUENCE {
3763     jmJobSubmissionID          OCTET STRING(SIZE(48)),
3764     jmJobIDJobSetIndex         Integer32 (0..32767),
3765     jmJobIDJobIndex            Integer32 (0..2147483647)
3766 }
3767

```



3768 jmJobSubmissionID OBJECT-TYPE  
3769     SYNTAX         OCTET STRING(SIZE(48))  
3770     MAX-ACCESS    not-accessible  
3771     STATUS         current  
3772     DESCRIPTION  
3773         "A quasi-unique 48-octet fixed-length string ID which  
3774         identifies the job within a particular client-server  
3775         environment. There are multiple formats for the  
3776         jmJobSubmissionID. Each format SHALL be uniquely identified.  
3777         See the JmJobSubmissionIDTypeTC textual convention. Each  
3778         format SHALL be registered using the procedures of a type 2  
3779         enum. See section 3.7.3 entitled: 'PWG Registration of Job  
3780         Submission Id Formats'.  
3781  
3782         If the requester (client or server) does not supply a job  
3783         submission ID in the job submission protocol, then the  
3784         recipient (server or device) SHALL assign a job submission ID  
3785         using any of the standard formats that have been reserved for  
3786         agents and adding the final 8 octets to distinguish the ID from  
3787         others submitted from the same requester.  
3788  
3789         The monitoring application, whether in the client or running  
3790         separately, MAY use the job submission ID to help identify  
3791         which jmJobIndex was assigned by the agent, i.e., in which row  
3792         the job information is in the other tables.  
3793  
3794         NOTE - fixed-length is used so that a management application  
3795         can use a shortened GetNext varbind (in SNMPv1 and SNMPv2) in  
3796         order to get the next submission ID, disregarding the remainder  
3797         of the ID in order to access jobs independent of the trailing  
3798         identifier part, e.g., to get all jobs submitted by a  
3799         particular jmJobOwner or submitted from a particular MAC  
3800         address."  
3801     REFERENCE  
3802         "See the JmJobSubmissionIDTypeTC textual convention.  
3803         See APPENDIX B - Support of Job Submission Protocols."  
3804      ::= { jmJobIDEntry 1 }  
3805

```
3806 jmJobIDJobSetIndex OBJECT-TYPE
3807     SYNTAX      Integer32 (0..32767)
3808     MAX-ACCESS  read-only
3809     STATUS      current
3810     DESCRIPTION
3811         "This object contains the value of the jmGeneralJobSetIndex for
3812         the job with the jmJobSubmissionID value, i.e., the job set
3813         index of the job set in which the job was placed when that
3814         server or device accepted the job. This 16-bit value in
3815         combination with the jmJobIDJobIndex value permits the
3816         management application to access the other tables to obtain the
3817         job-specific objects for this job."
3818     REFERENCE
3819         "See jmGeneralJobSetIndex in the jmGeneralTable."
3820     DEFVAL      { 0 }      -- 0 indicates no job set index
3821     ::= { jmJobIDEntry 2 }
3822
3823
3824
3825 jmJobIDJobIndex OBJECT-TYPE
3826     SYNTAX      Integer32 (0..2147483647)
3827     MAX-ACCESS  read-only
3828     STATUS      current
3829     DESCRIPTION
3830         "This object contains the value of the jmJobIndex for the job
3831         with the jmJobSubmissionID value, i.e., the job index for the
3832         job when the server or device accepted the job. This value, in
3833         combination with the jmJobIDJobSetIndex value, permits the
3834         management application to access the other tables to obtain the
3835         job-specific objects for this job."
3836     REFERENCE
3837         "See jmJobIndex in the jmJobTable."
3838     DEFVAL      { 0 }      -- 0 indicates no jmJobIndex value.
3839     ::= { jmJobIDEntry 3 }
3840
3841
3842
3843
```

```

3844 -- The Job Group (MANDATORY)
3845
3846 -- The jmJobGroup consists entirely of the jmJobTable.
3847
3848 jmJob OBJECT IDENTIFIER ::= { jobmonMIBObjects 3 }
3849
3850 jmJobTable OBJECT-TYPE
3851     SYNTAX      SEQUENCE OF JmJobEntry
3852     MAX-ACCESS  not-accessible
3853     STATUS      current
3854     DESCRIPTION
3855         "The jmJobTable consists of basic job state and status
3856         information for each job in a job set that (1) monitoring
3857         applications need to be able to access in a single SNMP Get
3858         operation, (2) that have a single value per job, and (3) that
3859         SHALL always be implemented."
3860     REFERENCE
3861         "The MANDATORY-GROUP macro specifies that this group is
3862         MANDATORY."
3863     ::= { jmJob 1 }
3864
3865
3866
3867 jmJobEntry OBJECT-TYPE
3868     SYNTAX      JmJobEntry
3869     MAX-ACCESS  not-accessible
3870     STATUS      current
3871     DESCRIPTION
3872         "Basic per-job state and status information.
3873
3874         An entry SHALL exist in this table for each job, no matter what
3875         the state of the job is. Each job SHALL appear in one and only
3876         one job set."
3877     REFERENCE
3878         "See Section 3.2 entitled 'The Job Tables'."
3879     INDEX { jmGeneralJobSetIndex, jmJobIndex }
3880     ::= { jmJobTable 1 }
3881
3882 JmJobEntry ::= SEQUENCE {
3883     jmJobIndex          Integer32 (1..2147483647),
3884     jmJobState          JmJobStateTC,
3885     jmJobStateReasons1 JmJobStateReasons1TC,
3886     jmNumberOfInterveningJobs Integer32 (-2..2147483647),
3887     jmJobKOctetsPerCopyRequested Integer32 (-2..2147483647),
3888     jmJobKOctetsProcessed Integer32 (-2..2147483647),
3889     jmJobImpressionsPerCopyRequested Integer32 (-2..2147483647),
3890     jmJobImpressionsCompleted Integer32 (-2..2147483647),
3891     jmJobOwner          JmJobStringTC (SIZE(0..63))
3892 }
3893

```

```
3894 jmJobIndex OBJECT-TYPE
3895     SYNTAX      Integer32 (1..2147483647)
3896     MAX-ACCESS  not-accessible
3897     STATUS      current
3898     DESCRIPTION
3899         "The sequential, monotonically increasing identifier index for
3900         the job generated by the server or device when that server or
3901         device accepted the job. This index value permits the
3902         management application to access the other tables to obtain the
3903         job-specific row entries."
3904     REFERENCE
3905         "See Section 3.2 entitled 'The Job Tables and the Oldest Active
3906         and Newest Active Indexes'.
3907         See Section 3.5 entitled 'Job Identification'.
3908         See also
3909
3910         jmGeneralNewestActiveJobIndex for the largest value of
3911         jmJobIndex.
3912         See JmJobSubmissionIDTypeTC for a limit on the size of this
3913         index if the agent represents it as an 8-digit decimal number."
3914     ::= { jmJobEntry 1 }
3915
3916
3917
3918 jmJobState OBJECT-TYPE
3919     SYNTAX      JmJobStateTC
3920     MAX-ACCESS  read-only
3921     STATUS      current
3922     DESCRIPTION
3923         "The current state of the job (pending, processing, completed,
3924         etc.). Agents SHALL implement only those states which are
3925         appropriate for the particular implementation. However,
3926         management applications SHALL be prepared to receive all the
3927         standard job states.
3928
3929         The final value for this object SHALL be one of: completed,
3930         canceled, or aborted. The minimum length of time that the
3931         agent SHALL maintain MIB data for a job in the completed,
3932         canceled, or aborted state before removing the job data from
3933         the jmJobIDTable and jmJobTable is specified by the value of
3934         the jmGeneralJobPersistence object."
3935     DEFVAL      { unknown }      -- default is unknown
3936     ::= { jmJobEntry 2 }
3937
```

```
3938 jmJobStateReasons1 OBJECT-TYPE
3939     SYNTAX      JmJobStateReasons1TC
3940     MAX-ACCESS  read-only
3941     STATUS      current
3942     DESCRIPTION
3943         "Additional information about the job's current state, i.e.,
3944         information that augments the value of the job's jmJobState
3945         object.
3946
3947         Implementation of any reason values is OPTIONAL, but an agent
3948         SHOULD return any reason information available. These values
3949         MAY be used with any job state or states for which the reason
3950         makes sense. Since the Job State Reasons will be more dynamic
3951         than the Job State, it is recommended that a job monitoring
3952         application read this object every time jmJobState is read.
3953         When the agent cannot provide a reason for the current state of
3954         the job, the value of the jmJobStateReasons1 object and
3955         jobStateReasonsN attributes SHALL be 0."
3956     REFERENCE
3957         "The jobStateReasonsN (N=2..4) attributes provide further
3958         additional information about the job's current state."
3959     DEFVAL      { 0 }          -- no reasons
3960     ::= { jmJobEntry 3 }
3961
3962
3963
3964 jmNumberOfInterveningJobs OBJECT-TYPE
3965     SYNTAX      Integer32 (-2..2147483647)
3966     MAX-ACCESS  read-only
3967     STATUS      current
3968     DESCRIPTION
3969         "The number of jobs that are expected to complete processing
3970         before this job has completed processing according to the
3971         implementation's queuing algorithm, if no other jobs were to be
3972         submitted. In other words, this value is the job's queue
3973         position. The agent SHALL return a value of 0 for this
3974         attribute when the job is the next job to complete processing
3975         (or has completed processing)."
3976     DEFVAL      { 0 }          -- default is no intervening jobs.
3977     ::= { jmJobEntry 4 }
3978
```

```
3979 jmJobKOctetsPerCopyRequested OBJECT-TYPE
3980     SYNTAX      Integer32 (-2..2147483647)
3981     MAX-ACCESS  read-only
3982     STATUS      current
3983     DESCRIPTION
3984         "The total size in K (1024) octets of the document(s) being
3985         requested to be processed in the job.  The agent SHALL round
3986         the actual number of octets up to the next highest K.  Thus 0
3987         octets SHALL be represented as '0', 1-1024 octets SHALL be
3988         represented as '1', 1025-2048 SHALL be represented as '2', etc.
3989
3990         In computing this value, the server/device SHALL not include
3991         the multiplicative factors contributed by (1) the number of
3992         document copies, and (2) the number of job copies, independent
3993         of whether the device can process multiple copies of the job or
3994         document without making multiple passes over the job or
3995         document data and independent of whether the output is collated
3996         or not.  Thus the server/device computation is independent of
3997         the implementation and indicates the size of the document(s)
3998         measured in K octets independent of the number of copies."
3999     DEFVAL      { -2 }      -- the default is unknown(-2)
4000     ::= { jmJobEntry 5 }
4001
4002
4003
4004 jmJobKOctetsProcessed OBJECT-TYPE
4005     SYNTAX      Integer32 (-2..2147483647)
4006     MAX-ACCESS  read-only
4007     STATUS      current
4008     DESCRIPTION
4009         "The total number of octets processed by the server or device
4010         measured in units of K (1024) octets so far.  The agent SHALL
4011         round the actual number of octets processed up to the next
4012         higher K.  Thus 0 octets SHALL be represented as '0', 1-1024
4013         octets SHALL be represented as '1', 1025-2048 octets SHALL be
4014         '2', etc.  For printing devices, this value is the number
4015         interpreted by the page description language interpreter rather
4016         than what has been marked on media.
4017
4018         For implementations where multiple copies are produced by the
4019         interpreter with only a single pass over the data, the final
4020         value SHALL be equal to the value of the
4021         jmJobKOctetsPerCopyRequested object.  For implementations where
4022         multiple copies are produced by the interpreter by processing
4023         the data for each copy, the final value SHALL be a multiple of
4024         the value of the jmJobKOctetsPerCopyRequested object.
4025
4026         NOTE - See the impressionsCompletedCurrentCopy and
4027         pagesCompletedCurrentCopy attributes for attributes that are
4028         reset on each document copy.
4029
```

4030 NOTE - The jmJobKOctetsProcessed object can be used with the  
 4031 jmJobKOctetsPerCopyRequested object to provide an indication of  
 4032 the relative progress of the job, provided that the  
 4033 multiplicative factor is taken into account for some  
 4034 implementations of multiple copies."  
 4035 DEFVAL { 0 } -- default is no octets processed.  
 4036 ::= { jmJobEntry 6 }  
 4037  
 4038  
 4039 jmJobImpressionsPerCopyRequested OBJECT-TYPE  
 4040 SYNTAX Integer32 (-2..2147483647)  
 4041 MAX-ACCESS read-only  
 4042 STATUS current  
 4043 DESCRIPTION  
 4044 "The total size in number of impressions of the document(s)  
 4045 submitted.  
 4046  
 4047 In computing this value, the server/device SHALL *not* include  
 4048 the multiplicative factors contributed by (1) the number of  
 4049 document copies, and (2) the number of job copies, independent  
 4050 of whether the device can process multiple copies of the job or  
 4051 document without making multiple passes over the job or  
 4052 document data and independent of whether the output is collated  
 4053 or not. Thus the server/device computation is independent of  
 4054 the implementation and reflects the size of the document(s)  
 4055 measured in impressions independent of the number of copies."  
 4056 REFERENCE  
 4057 "See the definition of the term 'impression' in Section 2."  
 4058 DEFVAL { -2 } -- default is unknown(-2)  
 4059 ::= { jmJobEntry 7 }  
 4060  
 4061  
 4062 jmJobImpressionsCompleted OBJECT-TYPE  
 4063 SYNTAX Integer32 (-2..2147483647)  
 4064 MAX-ACCESS read-only  
 4065 STATUS current  
 4066 DESCRIPTION  
 4067 "The total number of impressions completed for this job so far.  
 4068 For printing devices, the impressions completed includes  
 4069 interpreting, marking, and stacking the output. For other  
 4070 types of job services, the number of impressions completed  
 4071 includes the number of impressions processed.  
 4072  
 4073 NOTE - See the impressionsCompletedCurrentCopy and  
 4074 pagesCompletedCurrentCopy attributes for attributes that are  
 4075 reset on each document copy.  
 4076  
 4077 NOTE - The jmJobImpressionsCompleted object can be used with  
 4078 the jmJobImpressionsPerCopyRequested object to provide an  
 4079 indication of the relative progress of the job, provided that  
 4080 the multiplicative factor is taken into account for some  
 4081 implementations of multiple copies."



```
4082     REFERENCE
4083         "See the definition of the term 'impression' in Section 2 and
4084         the counting example in Section 3.4 entitled 'Monitoring Job
4085         Progress'."
4086     DEFVAL      { 0 }          -- default is no octets
4087     ::= { jmJobEntry 8 }
4088
4089
4090
4091 jmJobOwner OBJECT-TYPE
4092     SYNTAX      JmJobStringTC_(SIZE(0..63))
4093     MAX-ACCESS  read-only
4094     STATUS      current
4095     DESCRIPTION
4096         "The coded character set name of the user that submitted the
4097         job.  The method of assigning this user name will be system
4098         and/or site specific but the method MUST insure that the name
4099         is unique to the network that is visible to the client and
4100         target device.
4101
4102         This value SHOULD be the most authenticated name of the user
4103         submitting the job."
4104     REFERENCE
4105         "See the OBJECT compliance macro for the minimum maximum length
4106         required for conformance."
4107     DEFVAL      { ''H }          -- empty string
4108     ::= { jmJobEntry 9 }
4109
4110
4111
4112
```

```
4113 -- The Attribute Group (MANDATORY)
4114
4115 -- The jmAttributeGroup consists entirely of the jmAttributeTable.
4116 --
4117 -- Implementation of the two objects in this group is MANDATORY.
4118 -- See Section 3.1 entitled 'Conformance Considerations'.
4119 -- An agent SHALL implement any attribute if (1) the server or device
4120 -- supports the functionality represented by the attribute and (2) the
4121 -- information is available to the agent.
4122
4123 jmAttribute OBJECT IDENTIFIER ::= { jobmonMIBObjects 4 }
4124
4125
4126
4127 jmAttributeTable OBJECT-TYPE
4128     SYNTAX          SEQUENCE OF JmAttributeEntry
4129     MAX-ACCESS      not-accessible
4130     STATUS          current
4131     DESCRIPTION
4132         "The jmAttributeTable SHALL contain attributes of the job and
4133         document(s) for each job in a job set.  Instead of allocating
4134         distinct objects for each attribute, each attribute is
4135         represented as a separate row in the jmAttributeTable."
4136     REFERENCE
4137         "The MANDATORY-GROUP macro specifies that this group is
4138         MANDATORY.  An agent SHALL implement any attribute if (1) the
4139         server or device supports the functionality represented by the
4140         attribute and (2) the information is available to the agent. "
4141     ::= { jmAttribute 1 }
4142
4143
4144
4145 jmAttributeEntry OBJECT-TYPE
4146     SYNTAX          JmAttributeEntry
4147     MAX-ACCESS      not-accessible
4148     STATUS          current
4149     DESCRIPTION
4150         "Attributes representing information about the job and
4151         document(s) or resources required and/or consumed.
4152
4153         Each entry in the jmAttributeTable is a per-job entry with an
4154         extra index for each type of attribute (jmAttributeTypeIndex)
4155         that a job can have and an additional index
4156         (jmAttributeInstanceIndex) for those attributes that can have
4157         multiple instances per job.  The jmAttributeTypeIndex object
4158         SHALL contain an enum type that indicates the type of attribute
4159         (see the JmAttributeTypeTC textual-convention).  The value of
4160         the attribute SHALL be represented in either the
4161         jmAttributeValueAsInteger or jmAttributeValueAsOctets objects,
4162         and/or both, as specified in the JmAttributeTypeTC textual-
4163         convention.
4164
```

4165 The agent SHALL create rows in the jmAttributeTable as the  
 4166 server or device is able to discover the attributes either from  
 4167 the job submission protocol itself or from the document PDL.  
 4168 As the documents are interpreted, the interpreter MAY discover  
 4169 additional attributes and so the agent adds additional rows to  
 4170 this table. As the attributes that represent resources are  
 4171 actually consumed, the usage counter contained in the  
 4172 jmAttributeValueAsInteger object is incremented according to  
 4173 the units indicated in the description of the JmAttributeTypeTC  
 4174 enum.

4176 The agent SHALL maintain each row in the jmJobTable for at  
 4177 least the minimum time after a job completes as specified by  
 4178 the jmGeneralAttributePersistence object.

4180 Zero or more entries SHALL exist in this table for each job in  
 4181 a job set."

4182 REFERENCE

4183 "See Section 3.3 entitled 'The Attribute Mechanism' for a  
 4184 description of the jmAttributeTable."

4185 INDEX { jmGeneralJobSetIndex, jmJobIndex, jmAttributeTypeIndex,  
 4186 jmAttributeInstanceIndex }  
 4187 ::= { jmAttributeTable 1 }

4189 JmAttributeEntry ::= SEQUENCE {  
 4190 jmAttributeTypeIndex JmAttributeTypeTC,  
 4191 jmAttributeInstanceIndex Integer32 (1..32767),  
 4192 jmAttributeValueAsInteger Integer32<sub>-</sub> (-2..2147483647),  
 4193 jmAttributeValueAsOctets OCTET STRING(SIZE(0..63))  
 4194 }  
 4195

```
4196 jmAttributeTypeIndex OBJECT-TYPE
4197     SYNTAX      JmAttributeTypeTC
4198     MAX-ACCESS  not-accessible
4199     STATUS      current
4200     DESCRIPTION
4201         "The type of attribute that this row entry represents.
4202
4203         The type MAY identify information about the job or document(s)
4204         or MAY identify a resource required to process the job before
4205         the job start processing and/or consumed by the job as the job
4206         is processed.
4207
4208         Examples of job attributes (i.e., apply to the job as a whole)
4209         that have only one instance per job include:
4210         jobCopiesRequested(90), documentCopiesRequested(92),
4211         jobCopiesCompleted(91), documentCopiesCompleted(93), while
4212         examples of job attributes that may have more than one instance
4213         per job include: documentFormatIndex(37), and
4214         documentFormat(38).
4215
4216         Examples of document attributes (one instance per document)
4217         include: fileName(34), and documentName(35).
4218
4219         Examples of required and consumed resource attributes include:
4220         pagesRequested(130), mediumRequested(170), pagesCompleted(131),
4221         and mediumConsumed(171), respectively."
4222     ::= { jmAttributeEntry 1 }
4223
4224
4225
4226 jmAttributeInstanceIndex OBJECT-TYPE
4227     SYNTAX      Integer32 (1..32767)
4228     MAX-ACCESS  not-accessible
4229     STATUS      current
4230     DESCRIPTION
4231         "A running 16-bit index of the attributes of the same type for
4232         each job.  For those attributes with only a single instance per
4233         job, this index value SHALL be 1.  For those attributes that
4234         are a single value per document, the index value SHALL be the
4235         document number, starting with 1 for the first document in the
4236         job.  Jobs with only a single document SHALL use the index
4237         value of 1.  For those attributes that can have multiple values
4238         per job or per document, such as documentFormatIndex(37) or
4239         documentFormat(38), the index SHALL be a running index for the
4240         job as a whole, starting at 1."
4241     ::= { jmAttributeEntry 2 }
4242
```

```
4243 jmAttributeValueAsInteger OBJECT-TYPE
4244     SYNTAX      Integer32 (-2..2147483647)
4245     MAX-ACCESS  read-only
4246     STATUS      current
4247     DESCRIPTION
4248         "The integer value of the attribute.  The value of the
4249         attribute SHALL be represented as an integer if the enum
4250         description in the JmAttributeTypeTC textual-convention
4251         definition has the tag: 'INTEGER:'.
```

4252

4253 Depending on the enum definition, this object value MAY be an
4254 integer, a counter, an index, or an enum, depending on the
4255 jmAttributeTypeIndex value. The units of this value are
4256 specified in the enum description.

4257

4258 For those attributes that are accumulating job consumption as
4259 the job is processed as specified in the JmAttributeTypeTC
4260 textual-convention, SHALL contain the final value after the job
4261 completes processing, i.e., this value SHALL indicate the total
4262 usage of this resource made by the job.

4263

4264 A monitoring application is able to copy this value to a
4265 suitable longer term storage for later processing as part of an
4266 accounting system.

4267

4268 Since the agent MAY add attributes representing resources to
4269 this table while the job is waiting to be processed or being
4270 processed, which can be a long time before any of the resources
4271 are actually used, the agent SHALL set the value of the
4272 jmAttributeValueAsInteger object to 0 for resources that the
4273 job has not yet consumed.

4274

4275 Attributes for which the concept of an integer value is
4276 meaningless, such as fileName(34), jobName, and
4277 processingMessage, do not have the 'INTEGER:' tag in the
4278 JmAttributeTypeTC definition and so an agent SHALL always
4279 return a value of '-1' to indicate 'other' for the value of the
4280 jmAttributeValueAsInteger object for these attributes.

4281

4282 For attributes which do have the 'INTEGER:' tag in the
4283 JmAttributeTypeTC definition, if the integer value is not (yet)
4284 known, the agent either (1) SHALL not materialize the row in
4285 the jmAttributeTable until the value is known or (2) SHALL
4286 return a '-2' to represent an 'unknown' counting integer value,
4287 a '0' to represent an 'unknown' index value, and a '2' to
4288 represent an 'unknown(2)' enum value."

```
4289     DEFVAL      { -2 }      -- default value is unknown(-2)
4290     ::= { jmAttributeEntry 3 }
```

4291

```
4292 jmAttributeValueAsOctets OBJECT-TYPE
4293     SYNTAX      OCTET STRING(SIZE(0..63))
4294     MAX-ACCESS  read-only
4295     STATUS      current
4296     DESCRIPTION
4297         "The octet string value of the attribute.  The value of the
4298         attribute SHALL be represented as an OCTET STRING if the enum
4299         description in the JmAttributeTypeTC textual-convention
4300         definition has the tag: 'OCTETS:'."
4301
4302         Depending on the enum definition, this object value MAY be a
4303         coded character set string (text), such as 'JmUTF8StringTC', or
4304         a binary octet string, such as 'DateAndTime'.
4305
4306         Attributes for which the concept of an octet string value is
4307         meaningless, such as pagesCompleted, do not have the tag
4308         'OCTETS:' in the JmAttributeTypeTC definition and so the agent
4309         SHALL always return a zero length string for the value of the
4310         jmAttributeValueAsOctets object.
4311
4312         For attributes which do have the 'OCTETS:' tag in the
4313         JmAttributeTypeTC definition, if the OCTET STRING value is not
4314         (yet) known, the agent either SHALL not materialize the row in
4315         the jmAttributeTable until the value is known or SHALL return a
4316         zero-length string."
4317     DEFVAL      { ''H }      -- empty string
4318     ::= { jmAttributeEntry 4 }
4319
```

```
4320 -- Notifications and Trapping
4321 -- Reserved for the future
4322
4323 jobmonMIBNotifications OBJECT IDENTIFIER ::= { jobmonMIB 2_ }
4324
4325
4326
4327 -- Conformance Information
4328
4329 jmMIBConformance OBJECT IDENTIFIER ::= { jobmonMIB 3 }
4330
4331
4332
4333 -- compliance statements
4334 jmMIBCompliance MODULE-COMPLIANCE
4335     STATUS current
4336     DESCRIPTION
4337         "The compliance statement for agents that implement the
4338         job monitoring MIB."
4339     MODULE -- this module
4340     MANDATORY-GROUPS {
4341         jmGeneralGroup, jmJobIDGroup, jmJobGroup, jmAttributeGroup }
4342
4343     OBJECT jmGeneralJobSetName
4344     SYNTAX JmUTF8StringTC (SIZE(0..8))
4345     DESCRIPTION
4346         "Only 8 octets maximum string length NEED be supported by the
4347         agent."
4348
4349     OBJECT jmJobOwner
4350     SYNTAX JmJobStringTC (SIZE(0..16))
4351     DESCRIPTION
4352         "Only 16 octets maximum string length NEED be supported by the
4353         agent."
4354
4355 -- There are no CONDITIONALLY MANDATORY or OPTIONAL groups.
4356
4357 ::= { jmMIBConformance 1 }
4358
```



```
4359 jmMIBGroups OBJECT IDENTIFIER ::= { jmMIBConformance 2 }
4360
4361 jmGeneralGroup OBJECT-GROUP
4362     OBJECTS {
4363         jmGeneralNumberOfActiveJobs,    jmGeneralOldestActiveJobIndex,
4364         jmGeneralNewestActiveJobIndex,  jmGeneralJobPersistence,
4365         jmGeneralAttributePersistence,  jmGeneralJobSetName}
4366     STATUS current
4367     DESCRIPTION
4368         "The general group."
4369     ::= { jmMIBGroups 1 }
4370
4371
4372
4373 jmJobIDGroup OBJECT-GROUP
4374     OBJECTS {
4375         jmJobIDJobSetIndex, jmJobIDJobIndex }
4376     STATUS current
4377     DESCRIPTION
4378         "The job ID group."
4379     ::= { jmMIBGroups 2 }
4380
4381
4382
4383 jmJobGroup OBJECT-GROUP
4384     OBJECTS {
4385         jmJobState, jmJobStateReasons1, jmNumberOfInterveningJobs,
4386         jmJobKOctetsPerCopyRequested, jmJobKOctetsProcessed,
4387         jmJobImpressionsPerCopyRequested, jmJobImpressionsCompleted,
4388         jmJobOwner }
4389     STATUS current
4390     DESCRIPTION
4391         "The job group."
4392     ::= { jmMIBGroups 3 }
4393
4394
4395
4396 jmAttributeGroup OBJECT-GROUP
4397     OBJECTS {
4398         jmAttributeValueAsInteger, jmAttributeValueAsOctets }
4399     STATUS current
4400     DESCRIPTION
4401         "The attribute group."
4402     ::= { jmMIBGroups 4 }
4403
4404
4405 END
```

## 4406 5. Appendix A - Implementing the Job Life Cycle

4407 The job object has well-defined states and client operations that  
4408 affect the transition between the job states. Internal server and  
4409 device actions also affect the transitions of the job between the job  
4410 states. These states and transitions are referred to as the job's *life*  
4411 *cycle*.

4412 Not all implementations of job submission protocols have all of the  
4413 states of the job model specified here. The job model specified here  
4414 is intended to be a superset of most implementations. It is the  
4415 purpose of the agent to map the particular implementation's job life  
4416 cycle onto the one specified here. The agent MAY omit any states not  
4417 implemented. Only the processing and completed states are required to  
4418 be implemented by an agent. However, a conforming management  
4419 application SHALL be prepared to accept any of the states in the job  
4420 life cycle specified here, so that the management application can  
4421 interoperate with any conforming agent.

4422 The job states are intended to be user visible. The agent SHALL make  
4423 these states visible in the MIB, but only for the subset of job states  
4424 that the implementation has. Some implementations MAY need to have  
4425 sub-states of these user-visible states. The jmJobStateReasons1 object  
4426 and the jobStateReasonsN (N=2..4) attributes can be used to represent  
4427 the sub-states of the jobs.

4428 Job states are intended to last a user-visible length of time in most  
4429 implementations. However, some jobs may pass through some states in  
4430 zero time in some situations and/or in some implementations.

4431 The job model does not specify how accounting and auditing is  
4432 implemented, except to assume that accounting and auditing logs are  
4433 separate from the job life cycle and last longer than job entries in  
4434 the MIB. Jobs in the completed, aborted, or canceled states are not  
4435 logs, since jobs in these states are accessible via SNMP protocol  
4436 operations and SHALL be removed from the Job Monitoring MIB tables  
4437 after a site-settable or implementation-defined period of time. An  
4438 accounting application MAY copy accounting information incrementally to  
4439 an accounting log as a job processes, or MAY be copied while the job is  
4440 in the canceled, aborted, or completed states, depending on  
4441 implementation. The same is true for auditing logs.

4442 The jmJobState object specifies the standard job states. The normal  
4443 job state transitions are shown in the state transition diagram  
4444 presented in Table 1.

## 4445 6. APPENDIX B - Support of Job Submission Protocols

4446 A companion PWG document, entitled "Job Submission Protocol Mapping  
4447 Recommendations for the Job Monitoring MIB" [protomap] contains the  
4448 recommended usage of each of the objects and attributes in this MIB  
4449 with a number of job submission protocols. In particular, which job  
4450 submission ID format should be used is indicated for each job  
4451 submission protocol.

4452 Some job submission protocols have support for the client to specify a  
4453 job submission ID. A second approach is to enhance the document format  
4454 to embed the job submission ID in the document data. This second  
4455 approach is independent of the job submission protocol. This appendix  
4456 lists some examples of these approaches.

4457 Some PJJ implementations wrap a banner page as a PJJ job around a job  
4458 submitted by a client. If this results in multiple job submission IDs,  
4459 the agent SHALL create multiple jmJobIDEntry rows in the jmJobIDTable  
4460 that each point to the same job entry in the job tables. See the  
4461 specification of the jmJobIDEntry.

## 4462 7. References

4463 [char-set--policy] Harald Avelstrand, "IETF Policy on Character Sets  
4464 and Language", June 1997. Latest draft: <draft-avelstrand-charset-  
4465 policy-00.txt>

4466 [GB2312] GB 2312-1980, "Chinese People's Republic of China (PRC) mixed  
4467 one byte and two byte coded character set"

4468 [hr-mib] P. Grillo, S. Waldbusser, "Host Resources MIB", RFC 1514,  
4469 September 1993

4470 [iana] J. Reynolds, and J. Postel, "Assigned Numbers", STD 2, RFC 1700,  
4471 ISI, October 1994.

4472 [IANA-charsets] Coded Character Sets registered by IANA and assigned an  
4473 enum value for use in the CodedCharSet textual convention imported from  
4474 the Printer MIB. See ftp://ftp.isi.edu/in-  
4475 notes/iana/assignments/character-sets

4476 [iana-media-types] IANA Registration of MIME media types (MIME content  
4477 types/subtypes). See ftp://ftp.isi.edu/in-notes/iana/assignments/

4478 [ISO-639] ISO 639:1988 (E/F) - Code for Representation of names of  
4479 languages - The International Organization for Standardization, 1st  
4480 edition, 1988.

4481 [ISO 646] ISO/IEC 646:1991, "Information technology -- ISO 7-bit coded  
4482 character set for information interchange", JTC1/SC2.

- 4483 [ISO 8859] ISO/IEC 8859-1:1987, "Information technology -- 8-bit single  
4484 byte coded graphic character sets - Part 1: Latin alphabet No. 1,  
4485 JTC1/SC2."
- 4486 [ISO 2022] ISO/IEC 2022:1994 - "Information technology -- Character  
4487 code structure and extension techniques", JTC1/SC2.
- 4488 [ISO-3166] ISO 3166:1988 (E/F) - Codes for representation of names of  
4489 countries - The International Organization for Standardization, 3rd  
4490 edition, 1988-08-15."
- 4491 [ISO-10646] ISO/IEC 10646-1:1993, "Information technology -- Universal  
4492 Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and  
4493 Basic Multilingual Plane, JTC1/SC2.
- 4494 [iso-dpa] ISO/IEC 10175 Document Printing Application (DPA). See  
4495 <ftp://ftp.pwg.org/pub/pwg/dpa/>
- 4496 [ipp-model] Internet Printing Protocol/1.0: Model and Semantics, work  
4497 in progress on the IETF standards track. See [draft-ietf-ipp-model-](draft-ietf-ipp-model-09.txt)  
4498 [09.txt](http://www.pwg.org/ipp/index.html). See also <http://www.pwg.org/ipp/index.html>
- 4499 [JIS X0208] JIS X0208-1990, "Japanese two byte coded character set."
- 4500 [mib-II] MIB-II, RFC 1213.
- 4501 [print-mib] Smith, R., Wright, F., Hastings, T., Zilles, S. and  
4502 Gyllenskog, J., "Printer MIB", RFC 1759, proposed IETF standard, March  
4503 1995. See also [~~draft-print-mib-draft~~].
- 4504 [print-mib-draft] Turner, R., "Printer MIB", work in progress, on the  
4505 standards track as a draft standard: <[draft-ietf-printmib-mib-info-](draft-ietf-printmib-mib-info-02.txt)  
4506 [02.txt](draft-ietf-printmib-mib-info-02.txt)>, October 15, 1997.
- 4507 [protomap] Bergman, R., "Job Submission Protocol Mapping  
4508 Recommendations for the Job Monitoring MIB," work in progress as an  
4509 informational RFC. See <<draft-bergman-printmib-job-protomap-01.txt>>,  
4510 January 12, 1998.
- 4511 [pwg] The Printer Working Group is a printer industry consortium open  
4512 to any individuals. For more information, access the PWG web page:  
4513 <http://www.pwg.org>
- 4514 [req-words] S. Bradner, "Keywords for use in RFCs to Indicate  
4515 Requirement Levels", RFC 2119, March 1997.
- 4516 [rfc 1738] Berners-Lee, T., Masinter, L., McCahill, M., "Uniform  
4517 Resource Locators (URL)", RFC 1738, December 1994.
- 4518 [RFC-1766] Avelstrand, H., "Tags for the Identification of Languages",  
4519 RFC 1766, March 1995.

4520 [rfc 2130] C. Weider, C. Preston, K. Simonsen, H. Alvestrand, R.  
4521 Atkinson, M. Crispin, and P. Svanberg, "The Report of the IAB Character  
4522 Set Workshop held 29 Feb-1 March, 1997", April 1997, RFC 2130.

4523 [SMIv2-SMI] J. Case, et al. "Structure of Management Information for  
4524 Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC  
4525 1902, January 1996.

4526 [SMIv2-TC] J. Case, et al. "Textual Conventions for Version 2 of the  
4527 Simple Network Management Protocol (SNMPv2)", RFC 1903, January 1996.

4528 [tipsi] IEEE 1284.1, Transport-independent Printer System Interface  
4529 (TIPSI).

4530 [URI-spec] Berners-Lee, T., Masinter, L., McCahill, M. , "Uniform  
4531 Resource Locators (URL)", RFC 1738, December, 1994.

4532 [US-ASCII] Coded Character Set - 7-bit American Standard Code for  
4533 Information Interchange, ANSI X3.4-1986.

4534 [UTF-8] F. Yergeau, "UTF-8, a transformation format of Unicode and ISO  
4535 10646", RFC 2044, October 1996.

4536 8. Author's Addresses

4537 Ron Bergman  
4538 Dataproducts Corp.  
4539 1757 Tapo Canyon Road  
4540 Simi Valley, CA 93063-3394

4541  
4542 Phone: 805-578-4421  
4543 Fax: 805-578-4001  
4544 Email: rbergman@dpc.com

4545  
4546  
4547 Tom Hastings  
4548 Xerox Corporation, ESAE-231  
4549 701 S. Aviation Blvd.  
4550 El Segundo, CA 90245  
4551  
4552 Phone: 310-333-6413  
4553 Fax: 310-333-5514  
4554 EMail: hastings@cp10.es.xerox.com

4555  
4556  
4557 Scott A. Isaacson  
4558 Novell, Inc.  
4559 122 E 1700 S  
4560 Provo, UT 84606

4561  
4562 Phone: 801-861-7366  
4563 Fax: 801-861-4025

4564 EMail: scott\_isaacson@novell.com

4565

4566

4567 Harry Lewis

4568 IBM Corporation

4569 6300 Diagonal Hwy

4570 Boulder, CO 80301

4571

4572 Phone: (303) 924-5337

4573 Fax:

4574 Email: harryl@us.ibm.com

4575

4576

4577 Send questions and comments to the Printer Working Group (PWG)  
4578 using the Job Monitoring Project (JMP) Mailing List: jmp@pwg.org

4579

4580 To learn how to subscribe, send email to: jmp-request@pwg.org

4581

4582 Implementers of this specification are encouraged to join the jmp  
4583 mailing list in order to participate in discussions on any  
4584 clarifications needed and registration proposals for additional  
4585 attributes and values being reviewed in order to achieve consensus.

4586

4587 For further information, access the PWG web page under "JMP":

4588

4589 <http://www.pwg.org/>

4590

4591 Other Participants:

4592 Chuck Adams - Tektronix

4593 Jeff Barnett - IBM

4594 Keith Carter, IBM Corporation

4595 Jeff Copeland - QMS

4596 Andy Davidson - Tektronix

4597 Roger deBry - IBM

4598 Mabry Dozier - QMS

4599 Lee Ferrel - Canon

4600 Steve Gebert - IBM

4601 Robert Herriot - Sun Microsystems Inc.

4602 Shige Kanemitsu - Kyocera

4603 David Kellerman - Northlake Software

4604 Rick Landau - Digital

4605 Pete Loya - HP

4606 Ray Lutz - Cognisys

4607 Jay Martin - Underscore

4608 Mike MacKay, Novell, Inc.

4609 Stan McConnell - Xerox

4610 Carl-Uno Manros, Xerox, Corp.

4611 Pat Nogay - IBM

4612 Bob Pentecost - HP

4613 Rob Rhoads - Intel

4614 David Roach - Unisys  
4615 Stuart Rowley - Kyocera  
4616 Hiroyuki Sato - Canon  
4617 Bob Setterbo - Adobe  
4618 Gail Songer, EFI  
4619 Mike Timperman - Lexmark  
4620 Randy Turner - Sharp  
4621 William Wagner - Digital Products  
4622 Jim Walker - Dazel  
4623 Chris Wellens - Interworking Labs  
4624 Rob Whittle - Novell  
4625 Don Wright - Lexmark  
4626 Lloyd Young - Lexmark  
4627 Atsushi Yuki - Kyocera  
4628 Peter Zehler, Xerox, Corp.



## 4629 9. INDEX

4630 This index includes the textual conventions, the objects, and the  
 4631 attributes. Textual conventions all start with the prefix: "JM" and  
 4632 end with the suffix: "TC". Objects all starts with the prefix: "jm"  
 4633 followed by the group name. Attributes are identified with enums, and  
 4634 so start with any lower case letter and have no special prefix.

4635		
4636	colorantConsumed	70
4637	colorantRequested	69
4638	deviceNameRequested	59
4639	documentCopiesCompleted	64
4640	documentCopiesRequested	64
4641	documentFormat	61
4642	documentFormatIndex	60
4643	documentName	60
4644	fileName	60
4645	finishing	63
4646	fullColorImpressionsCompleted	66
4647	highlightColorImpressionsCompleted	67
4648	impressionsCompletedCurrentCopy	66
4649	impressionsInterpreted	66
4650	impressionsSentToDevice	66
4651	impressionsSpooled	66
4652	jmAttributeInstanceIndex	100
4653	jmAttributeTypeIndex	100
4654	JmAttributeTypeTC	53
4655	jmAttributeValueAsInteger	101
4656	jmAttributeValueAsOctets	102
4657	JmBooleanTC	43
4658	JmFinishingTC	41
4659	jmGeneralAttributePersistence	87
4660	jmGeneralJobPersistence	87
4661	jmGeneralJobSetIndex	85
4662	jmGeneralJobSetName	88
4663	jmGeneralNewestActiveJobIndex	86
4664	jmGeneralNumberOfActiveJobs	85
4665	jmGeneralOldestActiveJobIndex	86
4666	jmJobIDJobIndex	91
4667	jmJobIDJobSetIndex	91
4668	jmJobImpressionsCompleted	96
4669	jmJobImpressionsPerCopyRequested	96
4670	jmJobIndex	93
4671	jmJobKOctetsPerCopyRequested	95
4672	jmJobKOctetsProcessed	95
4673	jmJobOwner	97
4674	JmJobServiceTypesTC	74
4675	JmJobSourcePlatformTypeTC	40
4676	jmJobState	93
4677	jmJobStateReasons1	94
4678	JmJobStateReasons1TC	75

4679	JmJobStateReasons2TC	79
4680	JmJobStateReasons3TC	83
4681	JmJobStateReasons4TC	83
4682	JmJobStateTC	50
4683	JmJobStringTC	39
4684	jmJobSubmissionID	90
4685	JmJobSubmissionIDTypeTC	45
4686	JmMediumTypeTC	43
4687	JmNaturalLanguageTagTC	39
4688	jmNumberOfInterveningJobs	94
4689	JmPrinterResolutionTC	42
4690	JmPrintQualityTC	42
4691	JmTimeStampTC	40
4692	JmTonerEconomyTC	43
4693	JmUTF8StringTC	39
4694	jobAccountName	56
4695	jobCodedCharSet	55
4696	jobCollationType	65
4697	jobComment	60
4698	jobCompletionTime	71
4699	jobCopiesCompleted	64
4700	jobCopiesRequested	64
4701	jobHold	62
4702	jobHoldUntil	62
4703	jobKOctetsTransferred	65
4704	jobName	57
4705	jobNaturalLanguageTag	56
4706	jobOriginatingHost	59
4707	jobPriority	61
4708	jobProcessAfterDateAndTime	62
4709	jobProcessingCPUTime	71
4710	jobServiceTypes	58
4711	jobSourceChannelIndex	58
4712	jobSourcePlatformType	58
4713	jobStartedBeingHeldTime	71
4714	jobStartedProcessingTime	71
4715	jobStateReasons2	54
4716	jobStateReasons3	54
4717	jobStateReasons4	54
4718	jobSubmissionTime	70
4719	jobSubmissionToServerTime	70
4720	jobURI	56
4721	mediumConsumed	69
4722	mediumRequested	69
4723	numberOfDocuments	59
4724	other	53
4725	outputBin	62
4726	pagesCompleted	67
4727	pagesCompletedCurrentCopy	68
4728	pagesRequested	67
4729	physicalDevice	59
4730	printerResolutionRequested	63

4731	printerResolutionUsed	63
4732	printQualityRequested	63
4733	printQualityUsed	63
4734	processingMessage	54
4735	processingMessageNaturalLangTag	55
4736	queueNameRequested	59
4737	serverAssignedJobName	57
4738	sheetCompletedCopyNumber	65
4739	sheetCompletedDocumentNumber	65
4740	sheetsCompleted	68
4741	sheetsCompletedCurrentCopy	68
4742	sheetsRequested	68
4743	sides	63
4744	submittingApplicationName	58
4745	submittingServerName	58
4746	tonerDensityRequested	63
4747	tonerDensityUsed	64
4748	tonerEcomonyRequested	63
4749	tonerEcomonyUsed	63
4750		