

1 INTERNET-DRAFT

R. Bergman
Dataproducts Corp.
T. Hastings (editor)
Xerox Corporation
S. Isaacson
Novell, Inc.
H. Lewis
IBM Corp.
February 20, 1999

10 Job Monitoring MIB - V2.0
11 <draft-ietf-printmib-job-monitor-v2-00.txt>

12
13 Status of this Memo

14 This document is an Internet-Draft and is in full conformance with
15 all provisions of Section 10 of [RFC2026]. Internet-Drafts are
16 working documents of the Internet Engineering Task Force (IETF),
17 its areas, and its working groups. Note that other groups may
18 also distribute working documents as Internet-Drafts.

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

24 The list of current Internet-Drafts can be accessed at
25 <http://www.ietf.org/ietf/lid-abstracts.txt>

26 The list of Internet-Draft Shadow Directories can be accessed as
27 <http://www.ietf.org/shadow.html>.

28 This Internet-Draft expires on August 20, 1999.

29
30 Copyright (C) The Internet Society (1999). All Rights Reserved.

31
32 Abstract

33 This document has been developed and approved by the Printer
34 Working Group (PWG) as a PWG standard. It is intended to be
35 distributed as an Informational RFC. This document provides a
36 printer industry standard SNMP MIB for (1) monitoring the status
37 and progress of print jobs (2) obtaining resource requirements
38 before a job is processed, (3) monitoring resource consumption
39 while a job is being processed and (4) collecting resource
40 accounting data after the completion of a job. This MIB is
41 intended to be implemented (1) in a printer or (2) in a server
42 that supports one or more printers. Use of the object set is not
43 limited to printing. However, support for services other than
44 printing is outside the scope of this Job Monitoring MIB. Future

45 extensions to this MIB may include, but are not limited to, fax
46 machines and scanners.

47			
48		TABLE OF CONTENTS	
49	1	INTRODUCTION	7
50		1.1 Types of Information in the MIB	7
51		1.2 Types of Job Monitoring Applications	9
52	2	TERMINOLOGY AND JOB MODEL	10
53		2.1 System Configurations for the Job Monitoring MIB	13
54		2.1.1 Configuration 1 - client-printer	13
55		2.1.2 Configuration 2 - client-server-printer - agent in the server	14
56			14
57		2.1.3 Configuration 3 - client-server-printer - client monitors	15
58		printer agent and server	15
59	3	MANAGED OBJECT USAGE	17
60		3.1 Conformance Considerations	17
61		3.1.1 Conformance Terminology	17
62		3.1.2 Agent Conformance Requirements	17
63		3.1.2.1 MIB II System Group objects	18
64		3.1.2.2 MIB II Interface Group objects	18
65		3.1.2.3 Printer MIB objects	18
66		3.1.3 Job Monitoring Application Conformance Requirements	18
67		3.2 The Job Tables and the Oldest Active and Newest Active Indexes	19
68		3.3 The Attribute Mechanism and the Attribute Table(s)	21
69		3.3.1 Conformance of Attribute Implementation	22
70		3.3.2 Useful, 'Unknown', and 'Other' Values for Objects and	
71		Attributes	22
72		3.3.3 Index Value Attributes	23
73		3.3.4 Data Sub-types and Attribute Naming Conventions	23
74		3.3.5 Single-Value (Row) Versus Multi-Value (MULTI-ROW)	
75		Attributes	24
76		3.3.6 Requested Objects and Attributes	24
77		3.3.7 Consumption Attributes	24
78		3.3.8 Attribute Specifications	25
79		3.3.9 Job State Reason bit definitions	45
80		3.3.9.1 JmJobStateReasons1TC specification	46
81		3.3.9.2 JmJobStateReasons2TC specification	50
82		3.3.9.3 JmJobStateReasons3TC specification	53
83		3.3.9.4 JmJobStateReasons4TC specification	54
84		3.4 Monitoring Job Progress	54
85		3.5 Job Identification	58

86	3.5.1	The Job Submission ID specifications	59
87	3.6	Internationalization Considerations	64
88	3.6.1	Text generated by the server or device	64
89	3.6.2	Text supplied by the job submitter	65
90	3.6.3	'DateAndTime' for representing the date and time	66
91	3.7	IANA and PWG Registration Considerations	66
92	3.7.1	PWG Registration of enums	66
93	3.7.1.1	Type 1 enumerations	67
94	3.7.1.2	Type 2 enumerations	67
95	3.7.1.3	Type 3 enumeration	67
96	3.7.2	PWG Registration of type 2 bit values	68
97	3.7.3	PWG Registration of Job Submission Id Formats	68
98	3.7.4	PWG Registration of MIME types/sub-types for document-formats	68
99			68
100	3.8	Security Considerations	68
101	3.8.1	Read-Write objects	68
102	3.8.2	Read-Only Objects In Other User's Jobs	69
103	3.9	Notifications	69
104	4	MIB SPECIFICATION	69
105		Textual conventions for this MIB module	71
106		JmUTF8StringTC	71
107		JmJobStringTC	71
108		JmNaturalLanguageTagTC	71
109		JmTimeStampTC	72
110		JmJobSourcePlatformTypeTC	72
111		JmFinishingTC	73
112		JmPrintQualityTC	74
113		JmPrinterResolutionTC	74
114		JmTonerEconomyTC	75
115		JmBooleanTC	75
116		JmMediumTypeTC	75
117		JmJobCollationTypeTC	77
118		JmJobSubmissionIDTypeTC	77
119		JmJobStateTC	78
120		JmAttributeTypeTC	81
121		JmJobServiceTypesTC	85
122		JmJobStateReasons1TC	86
123		JmJobStateReasons2TC	86
124		JmJobStateReasons3TC	87
125		JmJobStateReasons4TC	87
126		The General Group (MANDATORY)	88
127		jmGeneralJobSetIndex (Int32(1..32767))	89
128		jmGeneralNumberOfActiveJobs (Int32(0..))	89
129		jmGeneralOldestActiveJobIndex (Int32(0..))	90

130	jmGeneralNewestActiveJobIndex	(Int32(0..))	90
131	jmGeneralJobPersistence	(Int32(15..))	91
132	jmGeneralAttributePersistence	(Int32(15..))	91
133	jmGeneralJobSetName	(UTF8String63)	92
134	The Job ID Group (MANDATORY)		93
135	jmJobSubmissionID	(OCTET STRING(SIZE(48)))	94
136	jmJobIDJobSetIndex	(Int32(0..32767))	95
137	jmJobIDJobIndex	(Int32(0..))	95
138	The Job Group (MANDATORY)		96
139	jmJobIndex	(Int32(1..))	97
140	jmJobState	(JmJobStateTC)	97
141	jmJobStateReasons1	(JmJobStateReasons1TC)	98
142	jmNumberOfInterveningJobs	(Int32(-2..))	98
143	jmJobKOctetsPerCopyRequested	(Int32(-2..))	99
144	jmJobKOctetsProcessed	(Int32(-2..))	99
145	jmJobImpressionsPerCopyRequested	(Int32(-2..))	100
146	jmJobImpressionsCompleted	(Int32(-2..))	100
147	jmJobOwner	(JobString63)	101
148	The Attribute Group (MANDATORY)		102
149	jmAttributeTypeIndex	(JmAttributeTypeTC)	104
150	jmAttributeInstanceIndex	(Int32(1..32767))	104
151	jmAttributeValueAsInteger	(Int32(-2..))	105
152	jmAttributeValueAsOctets	(Octets63)	106
153	The Mirror Attribute Group (OPTIONAL)		107
154	jmMirrorAttrTypeIndex	(JmAttributeTypeTC)	108
155	jmMirrorAttrInstanceIndex	(Int32(1..32767))	108
156	jmMirrorAttrValueAsInteger	(Int32(-2..))	109
157	jmMirrorAttrValueAsOctets	(Octets63)	109
158	The System Group (MANDATORY)		110
159	jmSystemVersionString	(JmUTF8StringTC)	110
160	jmSystemOptionSupport	(Int32(0..))	110
161	jmSystemAttrIntegerSupport	(Octets63)	111
162	jmSystemAttrOctetsSupport	(Octets63)	112
163	jmSystemAttrMultiRowSupport	(Octets63)	112
164	5	APPENDIX A - IMPLEMENTING THE JOB LIFE CYCLE	117
165	6	APPENDIX B - SUPPORT OF JOB SUBMISSION PROTOCOLS	118
166	7	REFERENCES	118
167	8	AUTHOR'S ADDRESSES	120
168	9	CHANGE HISTORY	123

169	9.1 Changes to produce version 2.0, dated February 20, 1999	123
170	9.2 Changes to produce version 1.2, dated October 2, 1998	124
171	9.3 Changes to produce version 1.1, dated October 1, 1998	124
172	10 INDEX	125
173		

174 Job Monitoring MIB

175 1 Introduction

176 This specification defines an official Printer Working Group (PWG)
177 [PWG] standard SNMP MIB for the monitoring of jobs on network printers.
178 This specification is being published as an IETF Information Document
179 for the convenience of the Internet community. In consultation with
180 the IETF Application Area Directors, it was concluded that this MIB
181 specification properly belongs as an Information document, because this
182 MIB monitors a service node on the network, rather than a network node
183 proper.

184 The Job Monitoring MIB is intended to be implemented by an agent within
185 a printer or the first server closest to the printer, where the printer
186 is either directly connected to the server only or the printer does not
187 contain the job monitoring MIB agent. It is recommended that
188 implementations place the SNMP agent as close as possible to the
189 processing of the print job. This MIB applies to printers with and
190 without spooling capabilities. This MIB is designed to be compatible
191 with most current commonly-used job submission protocols. In most
192 environments that support high function job submission/job control
193 protocols, like ISO DPA[iso-dpa], those protocols would be used to
194 monitor and manage print jobs rather than using the Job Monitoring MIB.

195 The Job Monitoring MIB consists of a General Group, a Job Submission ID
196 Group, a Job Group, and an Attribute Group. Each group is a table.
197 All accessible objects are read-only. The General Group contains
198 general information that applies to all jobs in a job set. The Job
199 Submission ID table maps the job submission ID that the client uses to
200 identify a job to the jmJobIndex that the Job Monitoring Agent uses to
201 identify jobs in the Job and Attribute tables. The Job table contains
202 the MANDATORY integer job state and status objects. The Attribute
203 table consists of multiple entries per job that specify (1) job and
204 document identification and parameters, (2) requested resources, and
205 (3) consumed resources during and after job processing/printing. A
206 larger number of job attributes are defined as textual conventions that
207 an agent SHALL return if the server or device implements the
208 functionality so represented and the agent has access to the
209 information. The Attribute table provides access to job attributes by
210 job index. An OPTIONAL Mirror Attribute table is defined which
211 provides access to the same job attributes by attribute. A MANDATORY
212 System Group provides a version number and objects that indicate which
213 options and attributes are supported.

214 **1.1 Types of Information in the MIB**

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

218 User:

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

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

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

227 Provide error and diagnostic information for jobs that did not
228 successfully complete.

229 Operator:

230 Provide a presentation of the state of all the jobs in the print
231 system.

232 Provide the ability to identify the user that submitted the print
233 job.

234 Provide the ability to identify the resources required by each
235 job.

236 Provide the ability to define which physical printers are
237 candidates for the print job.

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

242 Capacity Planner:

243 Provide the ability to determine printer utilization as a
244 function of time.

245 Provide the ability to determine how long jobs wait before
246 starting to print.

247 Accountant:

248 Provide information to allow the creation of a record of
249 resources consumed and printer usage data for charging users or
250 groups for resources consumed.

251 Provide information to allow the prediction of consumable usage
252 and resource need.

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

259 1.2 Types of Job Monitoring Applications

260 The Job Monitoring MIB is designed for the following types of
261 monitoring applications:

- 262 1. Monitor a single job starting when the job is submitted and
263 ending a defined period after the job completes. The Job
264 Submission ID table provides the map to find the specific job
265 to be monitored.
- 266 2. Monitor all 'active' jobs in a queue, which this specification
267 generalizes to a "job set". End users may use such a program
268 when selecting a least busy printer, so the MIB is designed for
269 such a program to start up quickly and find the information
270 needed quickly without having to read all (completed) jobs in
271 order to find the active jobs. System operators may also use
272 such a program, in which case it would be running for a long
273 period of time and may also be interested in the jobs that have
274 completed. Finally such a program may be used to provide an
275 enhanced console and logging capability.
- 276 3. Collect resource usage for accounting or system utilization
277 purposes that copy the completed job statistics to an
278 accounting system. It is recognized that depending on
279 accounting programs to copy MIB data during the job-retention
280 period is somewhat unreliable, since the accounting program may
281 not be running (or may have crashed). Such a program is also
282 expected to keep a shadow copy of the entire Job Attribute
283 table including completed, canceled, and aborted jobs which the
284 program updates on each polling cycle. Such a program polls at
285 the rate of the persistence of the Attribute table. The design
286 is not optimized to help such an application determine which
287 jobs are completed, canceled, or aborted. Instead, the
288 application SHOULD query each job that the application's shadow
289 copy shows was not complete, canceled, or aborted at the
290 previous poll cycle to see if it is now complete or canceled,
291 plus any new jobs that have been submitted.

292 The MIB provides a set of objects that represent a compatible subset of
293 job and document attributes of the ISO DPA standard[iso-dpa] and the
294 Internet Printing Protocol (IPP)[ipp-model], so that coherence is
295 maintained between these two protocols and the information presented to
296 end users and system operators by monitoring applications. However,
297 the job monitoring MIB is intended to be used with printers that
298 implement other job submitting and management protocols, such as IEEE
299 1284.1 (TIPSI)[tipsi], as well as with ones that do implement ISO DPA.

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

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

305 2 Terminology and Job Model

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

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

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

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

321 Attribute: A name, value-pair that specifies a job or document
322 instruction, a status, or a condition of a job or a document that has
323 been submitted to a server or device. A particular attribute NEED NOT
324 be present in each job instance. In other words, attributes are
325 present in a job instance only when there is a need to express the
326 value, either because (1) the client supplied a value in the job
327 submission protocol, (2) the document data contained an embedded
328 attribute, or (3) the server or device supplied a default value. An
329 agent MAY represent an attribute as an entry (row) in the Attribute
330 table in this MIB in which entries are present only when necessary.
331 Attributes are identified in this MIB by an enum.

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

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

- 342 Document: A sub-section within a job that contains print data and
343 *document instructions* that apply to just the document.
- 344 Document Instruction: An instruction specifying how to process the
345 document. Document instructions MAY be passed in the job submission
346 protocol separate from the actual document data, or MAY be embedded in
347 the document data or a combination, depending on the job submission
348 protocol and implementation.
- 349 End User: A user that uses a client to submit a print job. See
350 "user".
- 351 Impression: For a print job, an impression is the passage of the
352 entire side of a sheet by the marker, whether or not any marks are made
353 and independent of the number of passes that the side makes past the
354 marker. Thus a four pass color process counts as a single impression,
355 as does highlight color. Impression counters count all kinds:
356 monochrome, highlight color, and full process color, while full color
357 counters only count full color impressions, and high light color
358 counters only count high light color impressions.
- 359 One-sided processing involves one impression per sheet. Two-sided
360 processing involves two impressions per sheet. If a two-sided document
361 has an odd number of pages, the last sheet still counts as two
362 impressions, if that sheet makes two passes through the marker or the
363 marker marks on both sides of a sheet in a single pass. Two-up
364 printing is the placement of two logical pages on one side of a sheet
365 and so is still a single impression. See "page" and "sheet".
- 366 NOTE - Since impressions include blank sides, it is suggested that
367 accounting application implementers consider charging for sheets,
368 rather than impressions, possibly using the value of the sides
369 attribute to select different charges for one-sided versus two-sided
370 printing, since some users may think that impressions don't include
371 blank sides.
- 372 Internal Collation: The production of the sheets for each document copy
373 performed within the printing device by making multiple passes over
374 either the source or an intermediate representation of the document.
- 375 Job: A unit of work whose results are expected together without
376 interjection of unrelated results. A job contains one or more
377 *documents*.
- 378 Job Accounting: The activity of a management application of accessing
379 the MIB and recording what happens to the job during and after the
380 processing of the job.

381 Job Instruction: An instruction specifying how, when, or where the job
382 is to be processed. Job instructions MAY be passed in the job
383 submission protocol or MAY be embedded in the document data or a
384 combination depending on the job submission protocol and
385 implementation.

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

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

394 Job Set: A group of jobs that are queued and scheduled together
395 according to a specified scheduling algorithm for a specified device or
396 set of devices. For implementations that embed the SNMP agent in the
397 device, the MIB job set normally represents *all* the jobs known to the
398 device, so that the implementation only implements a single job set.
399 If the SNMP agent is implemented in a server that controls one or more
400 devices, each MIB job set represents a job queue for (1) a specific
401 device or (2) set of devices, if the server uses a single queue to load
402 balance between several devices. Each job set is disjoint; no job
403 SHALL be represented in more than one MIB job set.

404 Monitor: Short for Job Monitoring Application.

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

408 Proxy: An agent that acts as a concentrator for one or more other
409 agents by accepting SNMP operations on the behalf of one or more other
410 agents, forwarding them on to those other agents, gathering responses
411 from those other agents and returning them to the original requesting
412 monitor.

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

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

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

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

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

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

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

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

432 Supervisor: A server that contains a control program that controls a
433 printer or other device. A supervisor is a client to the printer or
434 other device.

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

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

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

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

441 **2.1 System Configurations for the Job Monitoring MIB**

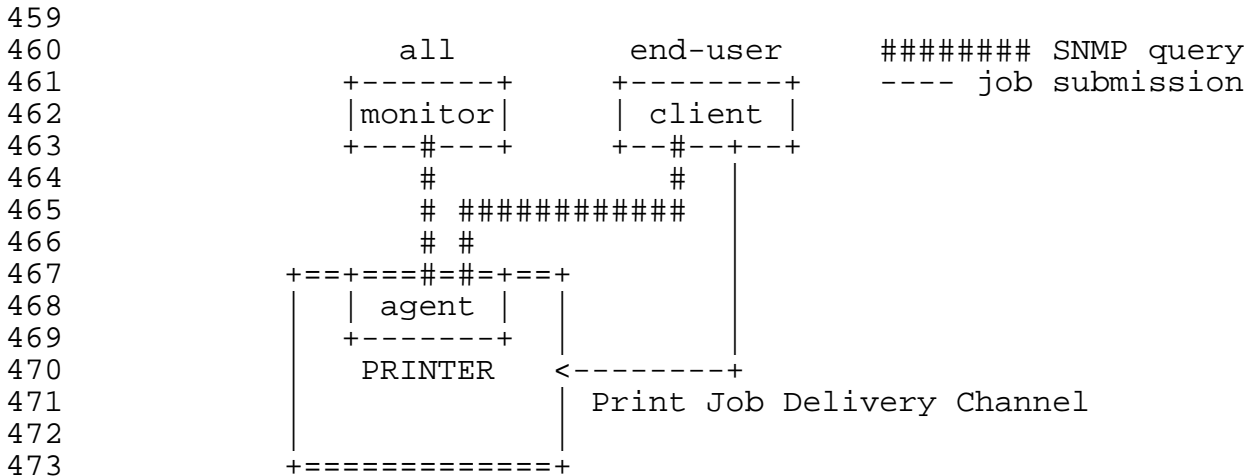
442 This section enumerates the three configurations in which the Job
443 Monitoring MIB is intended to be used. To simplify the pictures, the
444 *devices* are shown as *printers*. See section 1.1 entitled "Types of
445 Information in the MIB".

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

449 2.1.1 Configuration 1 - client-printer

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

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



474 Figure 2-1 - Configuration 1 - client-printer - agent in the printer

475 The Job Monitoring MIB is designed to support the following
 476 relationships (not shown in Figure 2-1):

- 477 1. Multiple clients MAY submit jobs to a printer.
- 478 2. Multiple clients MAY monitor a printer.
- 479 3. Multiple monitors MAY monitor a printer.
- 480 4. A client MAY submit jobs to multiple printers.
- 481 5. A monitor MAY monitor multiple printers.

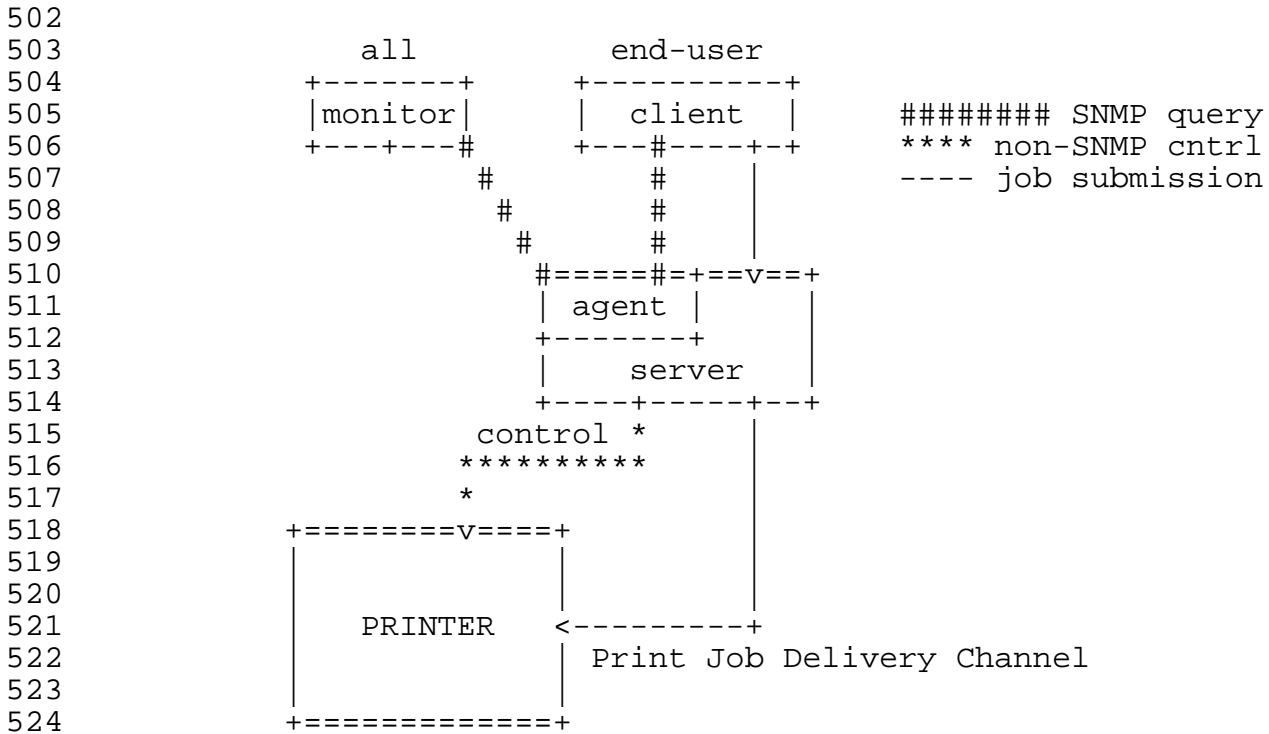
482 2.1.2 Configuration 2 - client-server-printer - agent in the server

483 In the client-server-printer configuration 2, the client(s) submit jobs
 484 to an intermediate server by some network connection, *not* directly to
 485 the printer. While configuration 2 is included, the design center for
 486 this MIB is configurations 1 and 3.

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

- 489 A Job Monitoring MIB agent that is part of the server (or a front
 490 for the server)

491 There is no SNMP Job Monitoring MIB agent in the printer in
 492 configuration 2, at least that the client or monitor are aware. In
 493 this configuration, the agent SHALL return the current values of the
 494 objects in the Job Monitoring MIB both for jobs the server keeps and
 495 jobs that the server has submitted to the printer. The Job Monitoring
 496 MIB agent obtains the required information from the printer by a method
 497 that is beyond the scope of this document. The agent in the server
 498 SHALL keep the job in the Job Monitoring MIB in the server as long as
 499 the job is in the printer, plus a defined time period after the job
 500 enters the completed state in which accounting programs can copy out
 501 the accounting data from the Job Monitoring MIB.



525 Figure 2-2 - Configuration 2 - client-server-printer - agent in the
 526 server

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

- 529 1. Multiple clients MAY submit jobs to a server.
- 530 2. Multiple clients MAY monitor a server.
- 531 3. Multiple monitors MAY monitor a server.
- 532 4. A client MAY submit jobs to multiple servers.
- 533 5. A monitor MAY monitor multiple servers.
- 534 6. Multiple servers MAY submit jobs to a printer.
- 535 7. Multiple servers MAY control a printer.

536 2.1.3 Configuration 3 - client-server-printer - client monitors printer
 537 agent and server

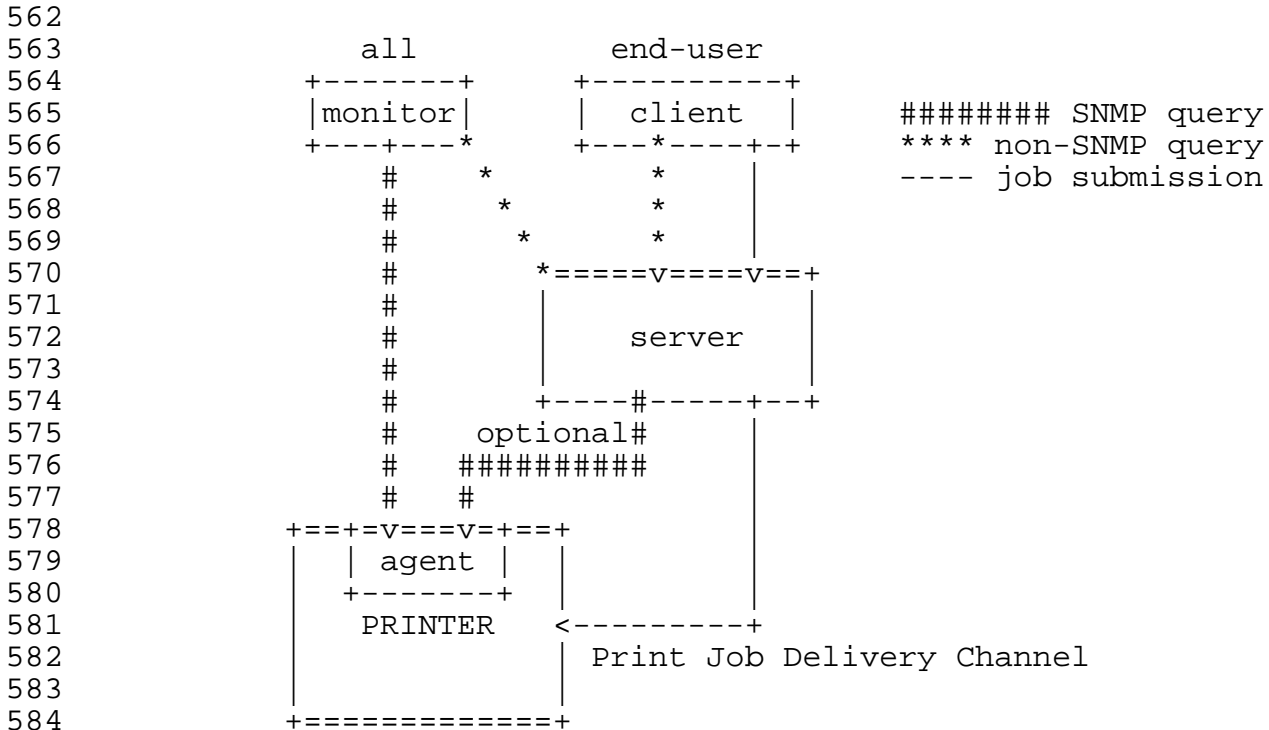
538 In the client-server-printer configuration 3, the client(s) submit jobs
 539 to an intermediate server by some network connection, *not* directly to
 540 the printer. That server does *not* contain a Job Monitoring MIB agent.

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

- 543 1. The server using some undefined protocol to monitor jobs in the
 544 server (that does not contain the Job Monitoring MIB) AND
- 545 2. A Job Monitoring MIB agent that is part of the printer to
 546 monitor jobs after the server passes the jobs to the printer.

547 In such configurations, the server deletes its copy of the job
 548 from the server after submitting the job to the printer usually
 549 almost immediately (before the job does much processing, if
 550 any).

551 In configuration 3, the agent (in the printer) SHALL keep the values of
 552 the objects in the Job Monitoring MIB that the agent implements updated
 553 for a job that the server has submitted to the printer. The agent
 554 SHALL obtain information about the jobs submitted to the printer from
 555 the server (either in the job submission protocol, in the document
 556 data, or by direct query of the server), in order to populate some of
 557 the objects the Job Monitoring MIB in the printer. The agent in the
 558 printer SHALL keep the job in the Job Monitoring MIB as long as the job
 559 is in the Printer, and longer in order to implement the completed state
 560 in which monitoring programs can copy out the accounting data from the
 561 Job Monitoring MIB.



585 Figure 2-3 - Configuration 3 - client-server-printer - client monitors
 586 printer agent and server

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

- 589 1. Multiple clients MAY submit jobs to a server.
- 590 2. Multiple clients MAY monitor a server.
- 591 3. Multiple monitors MAY monitor a server.
- 592 4. A client MAY submit jobs to multiple servers.
- 593 5. A monitor MAY monitor multiple servers.
- 594 6. Multiple servers MAY submit jobs to a printer.
- 595 7. Multiple servers MAY control a printer.

596 3 Managed Object Usage

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

598 **3.1 Conformance Considerations**

599 In order to achieve interoperability between job monitoring
600 applications and job monitoring agents, this specification includes the
601 conformance requirements for both monitoring applications and agents.

602 3.1.1 Conformance Terminology

603 This specification uses the verbs: "SHALL", "SHOULD", "MAY", and "NEED
604 NOT" to specify conformance requirements according to RFC 2119
605 [RFC2119] as follows:

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

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

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

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

618 3.1.2 Agent Conformance Requirements

619 A conforming agent:

- 620 1. SHALL implement *all* MANDATORY groups in this specification.
- 621 2. SHALL implement any attributes if (1) the server or device
622 supports the functionality represented by the attribute and (2)
623 the information is available to the agent.
- 624 3. SHOULD implement both forms of an attribute if it implements an
625 attribute that permits a choice of INTEGER and OCTET STRING
626 forms, since implementing both forms may help management
627 applications by giving them a choice of representations, since
628 the representation are equivalent. See the JmAttributeTypeTC
629 textual-convention.

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

632 3.1.2.1 MIB II System Group objects

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

636 3.1.2.2 MIB II Interface Group objects

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

640 3.1.2.3 Printer MIB objects

641 If the agent is providing access to a device that is a printer, the
642 agent SHALL implement all of the MANDATORY objects in the Printer
643 MIB[print-mib] and all the objects in other MIBs that conformance to
644 the Printer MIB requires, such as the Host Resources MIB[hr-mib]. If
645 the agent is providing access to a server that controls one or more
646 direct-connect or networked printers, the agent NEED NOT implement the
647 Printer MIB and NEED NOT implement the Host Resources MIB.

648 3.1.3 Job Monitoring Application Conformance Requirements

649 A conforming job monitoring application:

- 650 1. SHALL accept the full syntactic range for all objects in all
651 MANDATORY groups and all MANDATORY attributes that are required
652 to be implemented by an agent according to Section 3.1.2 and
653 SHALL either present them to the user or ignore them.
- 654 2. SHALL accept the full syntactic range for *all* attributes,
655 including enum and bit values specified in this specification
656 and additional ones that may be registered with the PWG and
657 SHALL either present them to the user or ignore them. In
658 particular, a conforming job monitoring application SHALL not
659 malfunction when receiving any standard or registered enum or
660 bit values. See Section 3.7 entitled "IANA and PWG
661 Registration Considerations".
- 662 3. SHALL NOT fail when operating with agents that materialize
663 attributes *after* the job has been submitted, as opposed to when
664 the job is submitted.
- 665 4. SHALL, if it supports a time attribute, accept either form of
666 the time attribute, since agents are free to implement either
667 time form.

668 3.2 The Job Tables and the Oldest Active and Newest Active Indexes

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

- 671 1. jmGeneralJobSetIndex - which job set
- 672 2. jmJobIndex - which job in the job set

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

- 676 1. jmGeneralOldestActiveJobIndex - the index of the active job
677 that has been in the tables the longest.
- 678 2. jmGeneralNewestActiveJobIndex - the index of the active job
679 that has been most recently added to the tables.

680 The agent SHALL assign the next incremental value of jmJobIndex to the
681 job, when a new job is accepted by the server or device to which the
682 agent is providing access. If the incremented value of jmJobIndex
683 would exceed the implementation-defined maximum value for jmJobIndex,
684 the agent SHALL 'wrap' back to 1. An agent uses the resulting value of
685 jmJobIndex for storing information in the jmJobTable and the
686 jmAttributeTable about the job.

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

692 It is recommended that agents that are providing access to
693 servers/devices that already allocate job-identifiers for jobs as
694 integers use the same integer value for the jmJobIndex. Then
695 management applications using this MIB and applications using other
696 protocols will see the same job identifiers for the same jobs. Agents
697 providing access to systems that contain jobs with a job identifier of
698 0 SHALL map the job identifier value 0 to a jmJobIndex value that is
699 one higher than the highest job identifier value that any job can have
700 on that system. Then only job 0 will have a different job-identifier
701 value than the job's jmJobIndex value.

702 NOTE - If a server or device accepts jobs using multiple job submission
703 protocols, it may be difficult for the agent to meet the recommendation
704 to use the job-identifier values that the server or device assigns as
705 the jmJobIndex value, unless the server/device assigns job-identifiers
706 for each of its job submission protocols from the same job-identifier
707 number space.

708 Each time a new job is accepted by the server or device that the agent
709 is providing access to AND that job is to be 'active' (pending,
710 processing, or processingStopped, but not pendingHeld), the agent SHALL
711 copy the value of the job's jmJobIndex to the
712 jmGeneralNewestActiveJobIndex object. If the new job is to be
713 'inactive' (pendingHeld state), the agent SHALL not change the value of
714 jmGeneralNewestActiveJobIndex object (though the agent SHALL assign the
715 next incremental jmJobIndex value to the job).

716 When a job transitions from one of the 'active' job states (pending,
717 processing, processingStopped) to one of the 'inactive' job states
718 (pendingHeld, completed, canceled, or aborted), with a jmJobIndex value
719 that matches the jmGeneralOldestActiveJobIndex object, the agent SHALL
720 advance (or wrap) the value to the next oldest 'active' job, if any.
721 See the JmJobStateTC textual-convention for a definition of the job
722 states.

723 Whenever a job transitions from one of the 'inactive' job states to one
724 of the 'active' job states (from pendingHeld to pending or processing),
725 the agent SHALL update the value of either the
726 jmGeneralOldestActiveJobIndex or the jmGeneralNewestActiveJobIndex
727 objects, or both, if the job's jmJobIndex value is outside the range
728 between jmGeneralOldestActiveJobIndex and
729 jmGeneralNewestActiveJobIndex.

730 When all jobs become 'inactive', i.e., enter the pendingHeld,
731 completed, canceled, or aborted states, the agent SHALL set the value
732 of both the jmGeneralOldestActiveJobIndex and
733 jmGeneralNewestActiveJobIndex objects to 0.

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

739 If an application detects that the jmGeneralNewestActiveJobIndex is
740 smaller than jmGeneralOldestActiveJobIndex, the job index has wrapped.
741 In this case, the application SHALL reset the index to 1 when the end
742 of the table is reached and continue the GetNext operations to find the
743 rest of the active jobs.

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

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

751 3.3 The Attribute Mechanism and the Attribute Table(s)

752 Attributes are similar to information objects, except that attributes
753 are identified by an enum, instead of an OID, so that attributes may be
754 registered without requiring a new MIB. Also an implementation that
755 does not have the functionality represented by the attribute can omit
756 the attribute entirely, rather than having to return a distinguished
757 value. The agent is free to materialize an attribute in the
758 jmAttributeTable as soon as the agent is aware of the value of the
759 attribute.

760 The agent materializes job attributes in a four-indexed
761 jmAttributeTable:

- 762 1. jmGeneralJobSetIndex - which job set
- 763 2. jmJobIndex - which job in the job set
- 764 3. jmAttributeTypeIndex - which attribute
- 765 4. jmAttributeInstanceIndex - which attribute instance for those
766 attributes that can have multiple values per job.

767 With this order of table indexing, an application can obtain all of the
768 attributes of a particular job using SNMPv1 GetNext or SNMPv2 GetBulk.

769 An OPTIONAL mirror table, called jmMirrorAttrTable, provides access to
770 the same job attributes, but with a different order to the indexes:

- 771 1. jmAttributeTypeIndex - which attribute
- 772 2. jmGeneralJobSetIndex - which job set
- 773 3. jmJobIndex - which job in the job set
- 774 4. jmAttributeInstanceIndex - which attribute instance for those
775 attributes that can have multiple values per job.

776 With this order of table indexing, an application can obtain selected
777 attributes of a number of jobs using SNMPv1 GetNext or SNMPv2 GetBulk.
778 A management application can determine whether or not this table is
779 implemented (even when the table is empty) by querying the
780 jmSystemOptionSupport object.

781 Some attributes represent information about a job, such as a file-name,
782 a document-name, a submission-time or a completion time. Other
783 attributes represent resources required, e.g., a medium or a colorant,
784 etc. to process the job before the job starts processing OR to indicate
785 the amount of the resource consumed during and after processing, e.g.,
786 pages completed or impressions completed. If both a required and a
787 consumed value of a resource is needed, this specification assigns two
788 separate attribute enums in the textual convention.

789 NOTE - The table of contents lists all the attributes in order. This
790 order is the order of enum assignments which is the order that the SNMP
791 GetNext operation returns attributes. Most attributes apply to all

792 three configurations covered by this MIB specification (see section 2.1
793 entitled "System Configurations for the Job Monitoring MIB"). Those
794 attributes that apply to a particular configuration are indicated as
795 'Configuration n:' and SHALL NOT be used with other configurations.

796 3.3.1 Conformance of Attribute Implementation

797 An agent SHALL implement any attribute if (1) the server or device
798 supports the functionality represented by the attribute and (2) the
799 information is available to the agent. The agent MAY create the
800 attribute row in the jmAttributeTable when the information is available
801 or MAY create the row earlier with the designated 'unknown' value
802 appropriate for that attribute. See next section.

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

806 3.3.2 Useful, 'Unknown', and 'Other' Values for Objects and Attributes

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

811 SNMP requires that if an object cannot be implemented because its
812 values cannot be accessed, then a compliant agent SHALL return an SNMP
813 error in SNMPv1 or an exception value in SNMPv2. However, this MIB has
814 been designed so that 'all' objects can and SHALL be implemented by an
815 agent, so that neither the SNMPv1 error nor the SNMPv2 exception value
816 SHALL be generated by the agent. This MIB has also been designed so
817 that when an agent materializes an attribute, the agent SHALL
818 materialize a row consisting of both the jmAttributeValueAsInteger and
819 jmAttributeValueAsOctets objects.

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

827 Since each attribute is represented by a row consisting of both the
828 jmAttributeValueAsInteger and jmAttributeValueAsOctets MANDATORY
829 objects, SNMP requires that the agent SHALL always create an attribute
830 row with both objects specified. However, for most attributes the
831 agent SHALL return a "useful" value for one of the objects and SHALL
832 return the 'other' value for the other object. For integer only
833 attributes, the agent SHALL always return a zero-length string value

834 for the jmAttributeValueAsOctets object. For octet string only
835 attributes, the agent SHALL always return a '-1' value for the
836 jmAttributeValueAsInteger object.

837 3.3.3 Index Value Attributes

838 A number of attributes are indexes in other tables. Such attribute
839 names end with the word 'Index'. If the agent has not (yet) assigned
840 an index value for a particular index attribute for a job, the agent
841 SHALL either: (1) return the value 0 or (2) not add this attribute to
842 the jmAttributeTable until the index value is assigned. In the
843 interests of brevity, the semantics for 0 is specified once here and is
844 not repeated for each index attribute specification and a DEFVAL of 0
845 is implied, even though the DEFVAL for jmAttributeValueAsInteger is -2.

846 3.3.4 Data Sub-types and Attribute Naming Conventions

847 Many attributes are sub-typed to give a more specific data type than
848 Integer32 or OCTET STRING. The data sub-type of each attribute is
849 indicated on the first line(s) of the description. Some attributes
850 have several different data sub-type representations. When an
851 attribute has both an Integer32 data sub-type and an OCTET STRING data
852 sub-type, the attribute can be represented in a single row in the
853 jmAttributeTable. In this case, the data sub-type name is not included
854 as the last part of the name of the attribute, e.g., documentFormat(38)
855 which is both an enum and/or a name. When the data sub-types cannot be
856 represented by a single row in the jmAttributeTable, each such
857 representation is considered a separate attribute and is assigned a
858 separate name and enum value. For these attributes, the name of the
859 data sub-type is the last part of the name of the attribute: Name,
860 Index, DateAndTime, TimeStamp, etc. For example,
861 documentFormatIndex(37) is an index.

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

866

'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.

867

868 3.3.5 Single-Value (Row) Versus Multi-Value (MULTI-ROW) Attributes

869 Most attributes have only one row per job. However, a few attributes
870 can have multiple values per job or even per document, where each value
871 is a separate row in the jmAttributeTable. Unless indicated with
872 'MULTI-ROW:' in the JmAttributeTypeTC description, an agent SHALL
873 ensure that each attribute occurs only once in the jmAttributeTable for
874 a job. Most of the 'MULTI-ROW' attributes do not allow duplicate
875 values, i.e., the agent SHALL ensure that each value occurs only once
876 for a job. Only if the specification of the 'MULTI-ROW' attribute also
877 says "There is no restriction on the same xxx occurring in multiple
878 rows" can the agent allow duplicate values to occur for the job.

879 NOTE - Duplicates are allowed for 'extensive' 'MULTI-ROW' attributes,
880 such as fileName(34) or documentName(35) which are specified to be
881 'per-document' attributes, but are *not* allowed for 'intensive' 'MULTI-
882 ROW' attributes, such as mediumConsumed(171) and documentFormat(38)
883 which are specified to be 'per-job' attributes.

884 3.3.6 Requested Objects and Attributes

885 A number of objects and attributes record requirements for the job.
886 Such object and attribute names end with the word 'Requested'. In the
887 interests of brevity, the phrase 'requested' means: (1) requested by
888 the client (or intervening server) in the job submission protocol and
889 may also mean (2) embedded in the submitted document data, and/or (3)
890 defaulted by the recipient device or server with the same semantics as
891 if the requester had supplied, depending on implementation. Also if a
892 value is supplied by the job submission client, and the server/device
893 determines a better value, through processing or other means, the agent
894 MAY return that better value for such object and attribute.

895 3.3.7 Consumption Attributes

896 A number of objects and attributes record consumption. Such attribute
897 names end with the word 'Completed' or 'Consumed'. If the job has not
898 yet consumed what that resource is metering, the agent either: (1)
899 SHALL return the value 0 or (2) SHALL *not* add this attribute to the
900 jmAttributeTable until the consumption begins. In the interests of
901 brevity, the semantics for 0 is specified once here and is *not* repeated
902 for each consumption attribute specification and a DEFVAL of 0 is
903 implied, even though the DEFVAL for jmAttributeValueAsInteger is -2.

904

905 3.3.8 Attribute Specifications

906 This section specifies the job attributes.

907 In the following definitions of the attributes, each description
908 indicates whether the useful value of the attribute SHALL be
909 represented using the jmAttributeValueAsInteger or the
910 jmAttributeValueAsOctets objects by the initial tag: 'INTEGER:' or
911 'OCTETS:', respectively.

912 Some attributes allow the agent implementer a choice of useful values
913 of either an integer, an octet string representation, or both,
914 depending on implementation. These attributes are indicated with
915 'INTEGER:' AND/OR 'OCTETS:' tags.

916 A very few attributes require both objects at the same time to
917 represent a pair of useful values (see mediumConsumed(171)). These
918 attributes are indicated with 'INTEGER:' AND 'OCTETS:' tags. See the
919 jmAttributeGroup for the descriptions of these two MANDATORY objects.

920 A management application can determine which attributes are supported
921 and whether the integer and/or the octet string values are supported
922 with meaningful value by querying the jmSystemAttrIntegerSupport and
923 jmSystemAttrOctetsSupport objects, respectively. Management
924 applications can also determine which supported attributes might
925 support more than one integer value or more than one octet string value
926 by querying jmSystemAttrMultiRowSupport.

927 These support bits are indicated in hex for each range in the line
928 starting with "support bits starting:". Note: these objects permit a
929 management application to determine the degree of support, even when
930 there are no jobs present in the system. They also permit management
931 middleware to fetch all attribute values for all jobs, including future
932 extensions, and keep them updated for one or more management
933 applications at the same time.

934 NOTE - The enum assignments are grouped logically with values assigned
935 in groups of 20, so that additional values may be registered in the
936 future and assigned a value that is part of their logical grouping.

937 Values in the range 2^{30} to $2^{31}-1$ are reserved for private or
938 experimental usage. This range corresponds to the same range reserved
939 in IPP. Implementers are warned that use of such values may conflict
940 with other implementations. Implementers are encouraged to request
941 registration of enum values following the procedures in Section 3.7.1.

942 NOTE: No attribute name exceeds 31 characters.

943 The standard attribute types are:

```

944
945     jmAttributeTypeIndex           Datatype
946     -----
947
948     other(1),                      Integer32 (-2..2147483647)
949                                     AND/OR
950                                     OCTET STRING(SIZE(0..63))
951     INTEGER:  and/or  OCTETS:  An attribute that is not in the
952     list and/or that has not been approved and registered with
953     the PWG.
954
955     ++++++
956     + Job State attributes (3 - 19 decimal)
957     +
958     + The following attributes specify the state of a job.
959     +       support bits starting: { '10'H }
960     ++++++
961
962     jobStateReasons2(3),            JmJobStateReasons2TC
963     INTEGER:  Additional information about the job's current
964     state that augments the jmJobState object.  See the
965     description under the JmJobStateReasons1TC textual-
966     convention.
967
968     jobStateReasons3(4),            JmJobStateReasons3TC
969     INTEGER:  Additional information about the job's current
970     state that augments the jmJobState object.  See the
971     description under JmJobStateReasons1TC textual-convention.
972
973     jobStateReasons4(5),            JmJobStateReasons4TC
974     INTEGER:  Additional information about the job's current
975     state that augments the jmJobState object.  See the
976     description under JmJobStateReasons1TC textual-convention.

```

977
978 processingMessage(6), JmUTF8StringTC (SIZE(0..63))
979 OCTETS: MULTI-ROW: A coded character set message that is
980 generated by the server or device during the processing of
981 the job as a simple form of processing log to show progress
982 and any problems. The natural language of each value is
983 specified by the corresponding
984 processingMessageNaturalLangTag(7) value.
985
986 NOTE - This attribute is intended for such conditions as
987 interpreter messages, rather than being the printable form
988 of the jmJobState and jmJobStateReasons1 objects and
989 jobStateReasons2, jobStateReasons3, and jobStateReasons4
990 attributes. In order to produce a localized printable form
991 of these job state objects/attribute, a management
992 application SHOULD produce a message from their enum and
993 bit values.
994
995 NOTE - There is no job description attribute in IPP/1.0
996 that corresponds to this attribute and this attribute does
997 not correspond to the IPP/1.0 'job-state-message' job
998 description attribute, which is just a printable form of
999 the IPP 'job-state' and 'job-state-reasons' job attributes.
1000
1001 There is no restriction for the same message occurring in
1002 multiple rows.
1003
1004 processingMessageNaturalLangTag(7), OCTET STRING(SIZE(0..63))
1005 OCTETS: MULTI-ROW: The natural language of the
1006 corresponding processingMessage(6) attribute value. See
1007 section 3.6.1, entitled 'Text generated by the server or
1008 device'.
1009
1010 If the agent does not know the natural language of the job
1011 processing message, the agent SHALL either (1) return a
1012 zero length string value for the
1013 processingMessageNaturalLangTag(7) attribute or (2) not
1014 return the processingMessageNaturalLangTag(7) attribute for
1015 the job.
1016
1017 There is no restriction for the same tag occurring in
1018 multiple rows, since when this attribute is implemented, it
1019 SHOULD have a value row for each corresponding
1020 processingMessage(6) attribute value row.

1021
1022 jobCodedCharSet(8), CodedCharSet
1023 INTEGER: The MIBenum identifier of the coded character set
1024 that the agent is using to represent coded character set
1025 objects and attributes of type 'JmJobStringTC'. These
1026 coded character set objects and attributes are either: (1)
1027 supplied by the job submitting client or (2) defaulted by
1028 the server or device when omitted by the job submitting
1029 client. The agent SHALL represent these objects and
1030 attributes in the MIB either (1) in the coded character set
1031 as they were submitted or (2) MAY convert the coded
1032 character set to another coded character set or encoding
1033 scheme as identified by the jobCodedCharSet(8) attribute.
1034 See section 3.6.2, entitled 'Text supplied by the job
1035 submitter'.
1036
1037 These MIBenum values are assigned by IANA [IANA-charsets]
1038 when the coded character sets are registered. The coded
1039 character set SHALL be one of the ones registered with IANA
1040 [IANA] and the enum value uses the CodedCharSet textual-
1041 convention from the Printer MIB. See the JmJobStringTC
1042 textual-convention.
1043
1044 If the agent does not know what coded character set was
1045 used by the job submitting client, the agent SHALL either
1046 (1) return the 'unknown(2)' value for the
1047 jobCodedCharSet(8) attribute or (2) not return the
1048 jobCodedCharSet(8) attribute for the job.
1049
1050 jobNaturalLanguageTag(9), OCTET STRING(SIZE(0..63))
1051 OCTETS: The natural language of the job attributes supplied
1052 by the job submitter or defaulted by the server or device
1053 for the job, i.e., all objects and attributes represented
1054 by the 'JmJobStringTC' textual-convention, such as jobName,
1055 mediumRequested, etc. See Section 3.6.2, entitled 'Text
1056 supplied by the job submitter'.
1057
1058 If the agent does not know what natural language was used
1059 by the job submitting client, the agent SHALL either (1)
1060 return a zero length string value for the
1061 jobNaturalLanguageTag(9) attribute or (2) not return
1062 jobNaturalLanguageTag(9) attribute for the job.
1063

```
1064 ++++++
1065 + Job Identification attributes (20 - 49 decimal)
1066 +
1067 + The following attributes help an end user, a system
1068 + operator, or an accounting program identify a job.
1069 +     support bits starting: { '000008'H }
1070 ++++++
1071
1072 jobURI(20),                               OCTET STRING(SIZE(0..63))
1073     OCTETS:  MULTI-ROW:  The job's Universal Resource
1074     Identifier (URI) [RFC1738].  See IPP [ipp-model] for
1075     example usage.
1076
1077     NOTE - The agent may be able to generate this value on each
1078     SNMP Get operation from smaller values, rather than having
1079     to store the entire URI.
1080
1081     If the URI exceeds 63 octets, the agent SHALL use multiple
1082     values, with the next 63 octets coming in the second value,
1083     etc.
1084
1085     NOTE - IPP [ipp-model] has a 1023-octet maximum length for
1086     a URI, though the URI standard itself and HTTP/1.1 specify
1087     no maximum length.
1088
1089 jobAccountName(21),                       OCTET STRING(SIZE(0..63))
1090     OCTETS:  Arbitrary binary information which MAY be coded
1091     character set data or encrypted data supplied by the
1092     submitting user for use by accounting services to allocate
1093     or categorize charges for services provided, such as a
1094     customer account name or number.
1095
1096     NOTE: This attribute NEED NOT be printable characters.
1097
1098 serverAssignedJobName(22),                 JmJobStringTC (SIZE(0..63))
1099     OCTETS:  Configuration 3 only:  The human readable string
1100     name, number, or ID of the job as assigned by the server
1101     that submitted the job to the device that the agent is
1102     providing access to with this MIB.
1103
1104     NOTE - This attribute is intended for enabling a user to
1105     find his/her job that a server submitted to a device when
1106     either the client does not support the jmJobSubmissionID or
1107     the server does not pass the jmJobSubmissionID through to
1108     the device.
```


1143
1144 jobServiceTypes(24), JmJobServiceTypesTC
1145 INTEGER: Specifies the type(s) of service to which the job
1146 has been submitted (print, fax, scan, etc.). The service
1147 type is bit encoded with each job service type so that more
1148 general and arbitrary services can be created, such as
1149 services with more than one destination type, or ones with
1150 only a source or only a destination. For example, a job
1151 service might scan, faxOut, and print a single job. In
1152 this case, three bits would be set in the jobServiceTypes
1153 attribute, corresponding to the hexadecimal values: 0x8 +
1154 0x20 + 0x4, respectively, yielding: 0x2C.
1155
1156 Whether this attribute is set from a job attribute supplied
1157 by the job submission client or is set by the recipient job
1158 submission server or device depends on the job submission
1159 protocol. This attribute SHALL be implemented if the
1160 server or device has other types in addition to or instead
1161 of printing.
1162
1163 One of the purposes of this attribute is to permit a
1164 requester to filter out jobs that are not of interest. For
1165 example, a printer operator may only be interested in jobs
1166 that include printing.
1167
1168 jobSourceChannelIndex(25), Integer32 (0..2147483647)
1169 INTEGER: The index of the row in the associated Printer
1170 MIB[print-mib] of the channel which is the source of the
1171 print job.
1172
1173 jobSourcePlatformType(26), JmJobSourcePlatformTypeTC
1174 INTEGER: The source platform type of the immediate
1175 upstream submitter that submitted the job to the server
1176 (configuration 2) or device (configuration 1 and 3) to
1177 which the agent is providing access. For configuration 1,
1178 this is the type of the client that submitted the job to
1179 the device; for configuration 2, this is the type of the
1180 client that submitted the job to the server; and for
1181 configuration 3, this is the type of the server that
1182 submitted the job to the device.
1183
1184 submittingServerName(27), JmJobStringTC (SIZE(0..63))
1185 OCTETS: For configuration 3 only: The administrative name
1186 of the server that submitted the job to the device.
1187
1188 submittingApplicationName(28), JmJobStringTC (SIZE(0..63))
1189 OCTETS: The name of the client application (not the server
1190 in configuration 3) that submitted the job to the server or
1191 device.

1192
1193 jobOriginatingHost(29), JmJobStringTC (SIZE(0..63))
1194 OCTETS: The name of the client host (not the server host
1195 name in configuration 3) that submitted the job to the
1196 server or device.
1197

1198 deviceNameRequested(30), JmJobStringTC (SIZE(0..63))
1199 OCTETS: The administratively defined coded character set
1200 name of the target device requested by the submitting user.
1201 For configuration 1, its value corresponds to the Printer
1202 MIB[print-mib]: prtGeneralPrinterName object. For
1203 configuration 2 and 3, its value is the name of the logical
1204 or physical device that the user supplied to indicate to
1205 the server on which device(s) they wanted the job to be
1206 processed.
1207

1208 queueNameRequested(31), JmJobStringTC (SIZE(0..63))
1209 OCTETS: The administratively defined coded character set
1210 name of the target queue requested by the submitting user.
1211 For configuration 1, its value corresponds to the queue in
1212 the device for which the agent is providing access. For
1213 configuration 2 and 3, its value is the name of the queue
1214 that the user supplied to indicate to the server on which
1215 device(s) they wanted the job to be processed.
1216

1217 NOTE - typically an implementation SHOULD support either
1218 the deviceNameRequested or queueNameRequested attribute,
1219 but not both.
1220

1221 physicalDevice(32), hrDeviceIndex
1222 AND/OR
1223 JmUTF8StringTC (SIZE(0..63))
1224 INTEGER: MULTI-ROW: The index of the physical device MIB
1225 instance requested/used, such as the Printer MIB[print-
1226 mib]. This value is an hrDeviceIndex value. See the Host
1227 Resources MIB[hr-mib].
1228

1229 AND/OR

1230

1231 OCTETS: MULTI-ROW: The name of the physical device to
1232 which the job is assigned.
1233

1234 numberOfDocuments(33), Integer32 (-2..2147483647)
1235 INTEGER: The number of documents in this job.
1236

1237 The agent SHOULD return this attribute if the job has more
1238 than one document.

1239
1240 fileName(34), JmJobStringTC (SIZE(0..63))
1241 OCTETS: MULTI-ROW: The coded character set file name or
1242 URI[URI-spec] of the document.
1243
1244 There is no restriction on the same file name occurring in
1245 multiple rows.
1246
1247 documentName(35), JmJobStringTC (SIZE(0..63))
1248 OCTETS: MULTI-ROW: The coded character set name of the
1249 document.
1250
1251 There is no restriction on the same document name occurring
1252 in multiple rows.
1253
1254 jobComment(36), JmJobStringTC (SIZE(0..63))
1255 OCTETS: An arbitrary human-readable coded character text
1256 string supplied by the submitting user or the job
1257 submitting application program for any purpose. For
1258 example, a user might indicate what he/she is going to do
1259 with the printed output or the job submitting application
1260 program might indicate how the document was produced.
1261
1262 The jobComment attribute is not intended to be a name; see
1263 the jobName attribute.
1264
1265 documentFormatIndex(37), Integer32 (0..2147483647)
1266 INTEGER: MULTI-ROW: The index in the prtInterpreterTable
1267 in the Printer MIB[print-mib] of the page description
1268 language (PDL) or control language interpreter that this
1269 job requires/uses. A document or a job MAY use more than
1270 one PDL or control language.
1271
1272 NOTE - As with all intensive attributes where multiple rows
1273 are allowed, there SHALL be only one distinct row for each
1274 distinct interpreter; there SHALL be no duplicates.
1275
1276 NOTE - This attribute type is intended to be used with an
1277 agent that implements the Printer MIB and SHALL not be used
1278 if the agent does not implement the Printer MIB. Such an
1279 agent SHALL use the documentFormat attribute instead.

```

1280
1281     documentFormat(38),                               PrtInterpreterLangFamilyTC
1282                                                         AND/OR
1283                                                         OCTET STRING(SIZE(0..63))
1284     INTEGER: MULTI-ROW: The interpreter language family
1285     corresponding to the Printer MIB[print-mib]
1286     prtInterpreterLangFamily object, that this job
1287     requires/uses. A document or a job MAY use more than one
1288     PDL or control language.
1289
1290     AND/OR
1291
1292     OCTETS: MULTI-ROW: The document format registered as a
1293     media type[iana-media-types], i.e., the name of the MIME
1294     content-type/subtype. Examples: 'application/postscript',
1295     'application/vnd.hp-PCL', 'application/pdf', 'text/plain'
1296     (US-ASCII SHALL be assumed), 'text/plain; charset=iso-8859-
1297     1', and 'application/octet-stream'. The IPP 'document-
1298     format' job attribute uses these same values with the same
1299     semantics. See the IPP [ipp-model] 'mimeType'
1300     attribute syntax and the document-format attribute for
1301     further examples and explanation.
1302
1303     ++++++
1304     + Job Parameter attributes (50 - 67 decimal)
1305     +
1306     + The following attributes represent input parameters
1307     + supplied by the submitting client in the job submission
1308     + protocol.
1309     +     support bits starting: { '00000000 000020'H }
1310     ++++++
1311
1312     jobPriority(50),                                   Integer32 (-2..100)
1313     INTEGER: The priority for scheduling the job. It is used
1314     by servers and devices that employ a priority-based
1315     scheduling algorithm.
1316
1317     A higher value specifies a higher priority. The value 1 is
1318     defined to indicate the lowest possible priority (a job
1319     which a priority-based scheduling algorithm SHALL pass over
1320     in favor of higher priority jobs). The value 100 is
1321     defined to indicate the highest possible priority.
1322     Priority is expected to be evenly or 'normally' distributed
1323     across this range. The mapping of vendor-defined priority
1324     over this range is implementation-specific. -2 indicates
1325     unknown.

```

1326
1327 jobProcessAfterDateAndTime(51), DateAndTime (SNMPv2-TC)
1328 OCTETS: The calendar date and time of day after which the
1329 job SHALL become a candidate to be scheduled for
1330 processing. If the value of this attribute is in the
1331 future, the server SHALL set the value of the job's
1332 jmJobState object to pendingHeld and add the
1333 jobProcessAfterSpecified bit value to the job's
1334 jmJobStateReasons1 object. When the specified date and
1335 time arrives, the server SHALL remove the
1336 jobProcessAfterSpecified bit value from the job's
1337 jmJobStateReasons1 object and, if no other reasons remain,
1338 SHALL change the job's jmJobState object to pending.
1339
1340 jobHold(52), JmBooleanTC
1341 INTEGER: If the value is 'true(4)', a client has
1342 explicitly specified that the job is to be held until
1343 explicitly released. Until the job is explicitly released
1344 by a client, the job SHALL be in the pendingHeld state with
1345 the jobHoldSpecified value in the jmJobStateReasons1
1346 attribute.
1347
1348 jobHoldUntil(53), JmJobStringTC (SIZE(0..63))
1349 OCTETS: The named time period during which the job SHALL
1350 become a candidate for processing, such as 'evening',
1351 'night', 'weekend', 'second-shift', 'third-shift', etc.,
1352 (supported values configured by the system administrator).
1353 See IPP [ipp-model] for the standard keyword values. Until
1354 that time period arrives, the job SHALL be in the
1355 pendingHeld state with the jobHoldUntilSpecified value in
1356 the jmJobStateReasons1 object. The value 'no-hold' SHALL
1357 indicate explicitly that no time period has been specified;
1358 the absence of this attribute SHALL indicate implicitly
1359 that no time period has been specified.
1360
1361 outputBin(54), Integer32 (0..2147483647)
1362 AND/OR
1363 JmJobStringTC (SIZE(0..63))
1364 INTEGER: MULTI-ROW: The output subunit index in the
1365 Printer MIB[print-mib]
1366
1367 AND/OR
1368
1369 OCTETS: MULTI-ROW: the name or number (represented as
1370 ASCII digits) of the output bin to which all or part of the
1371 job is placed in.
1372
1373 sides(55), Integer32 (-2..2)
1374 INTEGER: MULTI-ROW: The number of sides, '1' or '2', that
1375 any document in this job requires/used.

```
1376
1377     finishing(56),                               JmFinishingTC
1378         INTEGER: MULTI-ROW: Type of finishing that any document
1379         in this job requires/used.
1380
1381
1382     ++++++
1383     + Image Quality attributes (requested and consumed) (70 - 87)
1384     +
1385     + For devices that can vary the image quality.
1386     +         support bits starting: { '00000000 00000000 02'H }
1387     ++++++
1388
1389     printQualityRequested(70),                     JmPrintQualityTC
1390         INTEGER: MULTI-ROW: The print quality selection requested
1391         for a document in the job for printers that allow quality
1392         differentiation.
1393
1394     printQualityUsed(71),                           JmPrintQualityTC
1395         INTEGER: MULTI-ROW: The print quality selection actually
1396         used by a document in the job for printers that allow
1397         quality differentiation.
1398
1399     printerResolutionRequested(72),                JmPrinterResolutionTC
1400         OCTETS: MULTI-ROW: The printer resolution requested for a
1401         document in the job for printers that support resolution
1402         selection.
1403
1404     printerResolutionUsed(73),                     JmPrinterResolutionTC
1405         OCTETS: MULTI-ROW: The printer resolution actually used
1406         by a document in the job for printers that support
1407         resolution selection.
1408
1409     tonerEcomonyRequested(74),                     JmTonerEcomonyTC
1410         INTEGER: MULTI-ROW: The toner economy selection requested
1411         for documents in the job for printers that allow toner
1412         economy differentiation.
1413
1414     tonerEcomonyUsed(75),                           JmTonerEcomonyTC
1415         INTEGER: MULTI-ROW: The toner economy selection actually
1416         used by documents in the job for printers that allow toner
1417         economy differentiation.
1418
1419     tonerDensityRequested(76)                      Integer32 (-2..100)
1420         INTEGER: MULTI-ROW: The toner density requested for a
1421         document in this job for devices that can vary toner
1422         density levels. Level 1 is the lowest density and level
1423         100 is the highest density level. Devices with a smaller
1424         range, SHALL map the 1-100 range evenly onto the
1425         implemented range.
```

1426
1427 tonerDensityUsed(77), Integer32 (-2..100)
1428 INTEGER: MULTI-ROW: The toner density used by documents
1429 in this job for devices that can vary toner density levels.
1430 Level 1 is the lowest density and level 100 is the highest
1431 density level. Devices with a smaller range, SHALL map the
1432 1-100 range evenly onto the implemented range.
1433
1434 ++++++
1435 + Job Progress attributes (requested and consumed) (90-109)
1436 +
1437 + Pairs of these attributes can be used by monitoring
1438 + applications to show an indication of relative progress
1439 + to users. See section 3.4, entitled:
1440 + **'Monitoring Job Progress'**.
1441 + support bits starting: { '00000000 00000000 00000020'H }
1442 ++++++
1443
1444 jobCopiesRequested(90), Integer32 (-2..2147483647)
1445 INTEGER: The number of copies of the entire job that are
1446 to be produced.
1447
1448 jobCopiesCompleted(91), Integer32 (-2..2147483647)
1449 INTEGER: The number of copies of the entire job that have
1450 been completed so far.
1451
1452 documentCopiesRequested(92), Integer32 (-2..2147483647)
1453 INTEGER: The total count of the number of document copies
1454 requested for the job as a whole. If there are documents
1455 A, B, and C, and document B is specified to produce 4
1456 copies, the number of document copies requested is 6 for
1457 the job.
1458
1459 This attribute SHALL be used only when a job has multiple
1460 documents. The jobCopiesRequested attribute SHALL be used
1461 when the job has only one document.
1462
1463 documentCopiesCompleted(93), Integer32 (-2..2147483647)
1464 INTEGER: The total count of the number of document copies
1465 completed so far for the job as a whole. If there are
1466 documents A, B, and C, and document B is specified to
1467 produce 4 copies, the number of document copies starts a 0
1468 and runs up to 6 for the job as the job processes.
1469
1470 This attribute SHALL be used only when a job has multiple
1471 documents. The jobCopiesCompleted attribute SHALL be used
1472 when the job has only one document.

1473
1474 jobKOctetsTransferred(94), Integer32 (-2..2147483647)
1475 INTEGER: The number of K (1024) octets transferred to the
1476 server or device to which the agent is providing access.
1477 This count is independent of the number of copies of the
1478 job or documents that will be produced, but it is only a
1479 measure of the number of bytes transferred to the server or
1480 device.
1481
1482 The agent SHALL round the actual number of octets
1483 transferred up to the next higher K. Thus 0 octets SHALL
1484 be represented as '0', 1-1024 octets SHALL BE represented
1485 as '1', 1025-2048 SHALL be '2', etc. When the job
1486 completes, the values of the jmJobKOctetsPerCopyRequested
1487 object and the jobKOctetsTransferred attribute SHALL be
1488 equal.
1489
1490 NOTE - The jobKOctetsTransferred can be used with the
1491 jmJobKOctetsPerCopyRequested object in order to produce a
1492 relative indication of the progress of the job for agents
1493 that do not implement the jmJobKOctetsProcessed object.
1494
1495 sheetCompletedCopyNumber(95), Integer32 (-2..2147483647)
1496 INTEGER: The number of the copy being stacked for the
1497 current document. This number starts at 0, is set to 1
1498 when the first sheet of the first copy for each document is
1499 being stacked and is equal to n where n is the nth sheet
1500 stacked in the current document copy. See section 3.4 ,
1501 entitled 'Monitoring Job Progress'.
1502
1503 sheetCompletedDocumentNumber(96), Integer32 (-2..2147483647)
1504 INTEGER: The ordinal number of the document in the job
1505 that is currently being stacked. This number starts at 0,
1506 increments to 1 when the first sheet of the first document
1507 in the job is being stacked, and is equal to n where n is
1508 the nth document in the job, starting with 1.
1509
1510 Implementations that only support one document jobs SHOULD
1511 NOT implement this attribute.
1512
1513 jobCollationType(97), JmJobCollationTypeTC
1514 INTEGER: The type of job collation. See also Section 3.4,
1515 entitled 'Monitoring Job Progress'.
1516

```
1517 ++++++
1518 + Impression attributes (110 - 129 decimal)
1519 +
1520 + See the definition of the terms 'impression', 'sheet',
1521 + and 'page' in Section 2.
1522 +
1523 + See also jmJobImpressionsPerCopyRequested and
1524 + jmJobImpressionsCompleted objects in the jmJobTable.
1525 + support bits starting: { '00000000 00000000 00000000 0002'H }
1526 ++++++
1527
1528 impressionsSpooled(110),          Integer32 (-2..2147483647)
1529     INTEGER: The number of impressions spooled to the server
1530     or device for the job so far.
1531
1532 impressionsSentToDevice(111),     Integer32 (-2..2147483647)
1533     INTEGER: The number of impressions sent to the device for
1534     the job so far.
1535
1536 impressionsInterpreted(112),      Integer32 (-2..2147483647)
1537     INTEGER: The number of impressions interpreted for the job
1538     so far.
1539
1540 impressionsCompletedCurrentCopy(113),
1541     Integer32 (-2..2147483647)
1542     INTEGER: The number of impressions completed by the device
1543     for the current copy of the current document so far. For
1544     printing, the impressions completed includes interpreting,
1545     marking, and stacking the output. For other types of job
1546     services, the number of impressions completed includes the
1547     number of impressions processed.
1548
1549     This value SHALL be reset to 0 for each document in the job
1550     and for each document copy.
1551
1552 fullColorImpressionsCompleted(114), Integer32 (-2..2147483647)
1553     INTEGER: The number of full color impressions completed by
1554     the device for this job so far. For printing, the
1555     impressions completed includes interpreting, marking, and
1556     stacking the output. For other types of job services, the
1557     number of impressions completed includes the number of
1558     impressions processed. Full color impressions are typically
1559     defined as those requiring 3 or more colorants, but this
1560     MAY vary by implementation. In any case, the value of this
1561     attribute counts by 1 for each side that has full color,
1562     not by the number of colors per side (and the other
1563     impression counters are incremented, except
1564     highlightColorImpressionsCompleted(115)).
```

1565
1566 highlightColorImpressionsCompleted(115),
1567 Integer32 (-2..2147483647)
1568 INTEGER: The number of highlight color impressions
1569 completed by the device for this job so far. For printing,
1570 the impressions completed includes interpreting, marking,
1571 and stacking the output. For other types of job services,
1572 the number of impressions completed includes the number of
1573 impressions processed. Highlight color impressions are
1574 typically defined as those requiring black plus one other
1575 colorant, but this MAY vary by implementation. In any
1576 case, the value of this attribute counts by 1 for each side
1577 that has highlight color (and the other impression counters
1578 are incremented, except
1579 fullColorImpressionsCompleted(114)).
1580
1581 ++++++
1582 + Page attributes (130 - 149 decimal)
1583 +
1584 + See the definition of 'impression', 'sheet', and 'page'
1585 + in Section 2.
1586 + support bits starting:
1587 + { '00000000 00000000 00000000 00000000 20'H }
1588 ++++++
1589
1590 pagesRequested(130), Integer32 (-2..2147483647)
1591 INTEGER: The number of logical pages requested by the job
1592 to be processed.
1593
1594 pagesCompleted(131), Integer32 (-2..2147483647)
1595 INTEGER: The number of logical pages completed for this
1596 job so far.
1597
1598 For implementations where multiple copies are produced by
1599 the interpreter with only a single pass over the data, the
1600 final value SHALL be equal to the value of the
1601 pagesRequested object. For implementations where multiple
1602 copies are produced by the interpreter by processing the
1603 data for each copy, the final value SHALL be a multiple of
1604 the value of the pagesRequested object.
1605
1606 NOTE - See the impressionsCompletedCurrentCopy and
1607 pagesCompletedCurrentCopy attributes for attributes that
1608 are reset on each document copy.
1609
1610 NOTE - The pagesCompleted object can be used with the
1611 pagesRequested object to provide an indication of the
1612 relative progress of the job, provided that the
1613 multiplicative factor is taken into account for some
1614 implementations of multiple copies.

1615
1616 pagesCompletedCurrentCopy(132), Integer32 (-2..2147483647)
1617 INTEGER: The number of logical pages completed for the
1618 current copy of the document so far. This value SHALL be
1619 reset to 0 for each document in the job and for each
1620 document copy.
1621
1622 ++++++
1623 + Sheet attributes (150 - 169 decimal)
1624 +
1625 + See the definition of 'impression', 'sheet', and 'page'
1626 + in Section 2.
1627 + support bits starting:
1628 + { '00000000 00000000 00000000 00000000 000002'H }
1629 ++++++
1630
1631 sheetsRequested(150), Integer32 (-2..2147483647)
1632 INTEGER: The total number of medium sheets requested to be
1633 produced for this job.
1634
1635 Unlike the jmJobKOctetsPerCopyRequested and
1636 jmJobImpressionsPerCopyRequested attributes, the
1637 sheetsRequested(150) attribute SHALL include the
1638 multiplicative factor contributed by the number of copies
1639 and so is the total number of sheets to be produced by the
1640 job, as opposed to the size of the document(s) submitted.
1641
1642 sheetsCompleted(151), Integer32 (-2..2147483647)
1643 INTEGER: The total number of medium sheets that have
1644 completed marking and stacking for the entire job so far
1645 whether those sheets have been processed on one side or on
1646 both.
1647
1648 sheetsCompletedCurrentCopy(152), Integer32 (-2..2147483647)
1649 INTEGER: The number of medium sheets that have completed
1650 marking and stacking for the current copy of a document in
1651 the job so far whether those sheets have been processed on
1652 one side or on both.
1653
1654 The value of this attribute SHALL be 0 before the job
1655 starts processing and SHALL be reset to 1 after the first
1656 sheet of each document and document copy in the job is
1657 processed and stacked.
1658

```

1659 ++++++
1660 + Resources attributes (requested and consumed) (170 - 189)
1661 +
1662 + Pairs of these attributes can be used by monitoring
1663 + applications to show an indication of relative usage to
1664 + users, i.e., a 'thermometer'.
1665 + support bits starting:
1666 + { '00000000 00000000 00000000 00000000 00000000 0020'H }
1667 ++++++
1668
1669 mediumRequested(170),                               JmMediumTypeTC
1670                                                         AND/OR
1671                                                         JmJobStringTC (SIZE(0..63))
1672     INTEGER:  MULTI-ROW:  The type
1673     AND/OR
1674     OCTETS:  MULTI-ROW:  the name of the medium that is
1675     required by the job.
1676
1677     NOTE - The name (JmJobStringTC) values correspond to the
1678     name values of the prtInputMediaName object in the Printer
1679     MIB [print-mib] and the name, size, and input tray values
1680     of the IPP 'media' attribute [ipp-model].
1681
1682 mediumConsumed(171),                               Integer32 (-2..2147483647)
1683                                                         AND
1684                                                         JmJobStringTC (SIZE(0..63))
1685     INTEGER:  MULTI-ROW:  The number of sheets
1686     AND
1687     OCTETS:  MULTI-ROW:  the name of the medium that has been
1688     consumed so far whether those sheets have been processed on
1689     one side or on both.
1690
1691     This attribute SHALL have both Integer32 and OCTET STRING
1692     (represented as JmJobStringTC) values.
1693
1694     NOTE - The name (JmJobStringTC) values correspond to the
1695     name values of the prtInputMediaName object in the Printer
1696     MIB [print-mib] and the name, size, and input tray values
1697     of the IPP 'media' attribute [ipp-model].
1698
1699 colorantRequested(172),                             Integer32 (-2..2147483647)
1700                                                         AND/OR
1701                                                         JmJobStringTC (SIZE(0..63))
1702     INTEGER:  MULTI-ROW:  The index (prtMarkerColorantIndex) in
1703     the Printer MIB[print-mib]
1704     AND/OR
1705     OCTETS:  MULTI-ROW:  the name of the colorant requested.
1706
1707     NOTE - The name (JmJobStringTC) values correspond to the
1708     name values of the prtMarkerColorantValue object in the
1709     Printer MIB.  Examples are: red, blue.

```

1710
1711 colorantConsumed(173), Integer32 (-2..2147483647)
1712 AND/OR
1713 JmJobStringTC (SIZE(0..63))
1714 INTEGER: MULTI-ROW: The index (prtMarkerColorantIndex) in
1715 the Printer MIB[print-mib]
1716 AND/OR
1717 OCTETS: MULTI-ROW: the name of the colorant consumed.
1718
1719 NOTE - The name (JmJobStringTC) values correspond to the
1720 name values of the prtMarkerColorantValue object in the
1721 Printer MIB. Examples are: red, blue
1722
1723 mediumTypeConsumed(174), Integer32 (-2..2147483647)
1724 AND
1725 JmJobStringTC (SIZE(0..63))
1726 INTEGER: MULTI-ROW: The number of sheets of the indicated
1727 medium type that has been consumed so far whether those
1728 sheets have been processed on one side or on both
1729 AND
1730 OCTETS: MULTI-ROW: the name of that medium type.
1731
1732 This attribute SHALL have both Integer32 and OCTET STRING
1733 (represented as JmJobStringTC) values.
1734
1735 NOTE - The type name (JmJobStringTC) values correspond to
1736 the type name values of the prtInputMediaType object in the
1737 Printer MIB [print-mib]. Values are: 'stationery',
1738 'transparency', 'envelope', etc. These medium type names
1739 correspond to the enum values of JmMediumTypeTC used in the
1740 mediumRequested attribute.
1741
1742 mediumSizeConsumed(175), Integer32 (-2..2147483647)
1743 AND
1744 JmJobStringTC (SIZE(0..63))
1745 INTEGER: MULTI-ROW: The number of sheets of the indicated
1746 medium size that has been consumed so far whether those
1747 sheets have been processed on one side or on both
1748 AND
1749 OCTETS: MULTI-ROW: the name of that medium size.
1750
1751 This attribute SHALL have both Integer32 and OCTET STRING
1752 (represented as JmJobStringTC) values.
1753
1754 NOTE - The size name (JmJobStringTC) values correspond to
1755 the size name values in the Printer MIB [print-mib]
1756 Appendix B. These size name values are also a subset of
1757 the keyword values defined by [ipp-model] for the 'media'
1758 Job Template attribute. Values are: 'letter', 'a', 'iso-
1759 a4', 'jis-b4', etc.
1760

```

1761 ++++++
1762 + Time attributes (set by server or device) (190 - 209 decimal)
1763 +
1764 + This section of attributes are ones that are set by the
1765 + server or device that accepts jobs. Two forms of time are
1766 + provided. Each form is represented in a separate attribute.
1767 + See section 3.1.2 and section 3.1.3 for the
1768 + conformance requirements for time attribute for agents and
1769 + monitoring applications, respectively. The two forms are:
1770 +
1771 + 'DateAndTime' is an 8 or 11 octet binary encoded year,
1772 + month, day, hour, minute, second, deci-second with
1773 + optional offset from UTC. See SNMPv2-TC [SMIV2-TC].
1774 +
1775 + NOTE: 'DateAndTime' is not printable characters; it is
1776 + binary.
1777 +
1778 + 'JmTimeStampTC' is the time of day measured in the number of
1779 + seconds since the system was booted.
1780 + support bits starting:
1781 + { '00000000 00000000 00000000 00000000 00000000 00000002'H }
1782 ++++++
1783
1784 jobSubmissionToServerTime(190),      JmTimeStampTC
1785                                     AND/OR
1786                                     DateAndTime
1787     INTEGER: Configuration 3 only: The time
1788     AND/OR
1789     OCTETS: the date and time that the job was submitted to
1790     the server (as distinguished from the device which uses
1791     jobSubmissionTime).
1792
1793 jobSubmissionTime(191),              JmTimeStampTC
1794                                     AND/OR
1795                                     DateAndTime
1796     INTEGER: Configurations 1, 2, and 3: The time
1797     AND/OR
1798     OCTETS: the date and time that the job was submitted to
1799     the server or device to which the agent is providing
1800     access.
1801
1802 jobStartedBeingHeldTime(192),        JmTimeStampTC
1803                                     AND/OR
1804                                     DateAndTime
1805     INTEGER: The time
1806     AND/OR
1807     OCTETS: the date and time that the job last entered the
1808     pendingHeld state. If the job has never entered the
1809     pendingHeld state, then the value SHALL be '0' or the
1810     attribute SHALL not be present in the table.

```

1811
1812 jobStartedProcessingTime(193), JmTimeStampTC
1813 AND/OR
1814 DateAndTime
1815 INTEGER: The time
1816 AND/OR
1817 OCTETS: the date and time that the job started processing.
1818
1819 jobCompletionTime(194), JmTimeStampTC
1820 AND/OR
1821 DateAndTime
1822 INTEGER: The time
1823 AND/OR
1824 OCTETS: the date and time that the job entered the
1825 completed, canceled, or aborted state.
1826
1827 jobProcessingCPUtime(195) Integer32 (-2..2147483647)
1828 UNITS 'seconds'
1829 INTEGER: The amount of CPU time in seconds that the job
1830 has been in the processing state. If the job enters the
1831 processingStopped state, that elapsed time SHALL not be
1832 included. In other words, the jobProcessingCPUtime value
1833 SHOULD be relatively repeatable when the same job is
1834 processed again on the same device.

1835 3.3.9 Job State Reason bit definitions

1836 The JmJobStateReasonsMTC ($N=1..4$) textual-conventions are used with the
1837 jmJobStateReasons1 object and jobStateReasonsN ($N=2..4$), respectively,
1838 to provide additional information regarding the current jmJobState
1839 object value. These values MAY be used with any job state or states
1840 for which the reason makes sense.

1841 NOTE - While values cannot be added to the jmJobState object without
1842 impacting deployed clients that take actions upon receiving jmJobState
1843 values, it is the intent that additional JmJobStateReasonsMTC enums can
1844 be defined and registered without impacting such deployed clients. In
1845 other words, the jmJobStateReasons1 object and jobStateReasonsN
1846 attributes are intended to be extensible.

1847 NOTE - The Job Monitoring MIB contains a superset of the IPP
1848 values[ipp-model] for the IPP 'job-state-reasons' attribute, since the
1849 Job Monitoring MIB is intended to cover other job submission protocols
1850 as well. Also some of the names of the reasons have been changed from
1851 'printer' to 'device', since the Job Monitoring MIB is intended to
1852 cover additional types of devices, including input devices, such as
1853 scanners.

1854 **3.3.9.1 JmJobStateReasons1TC specification**

1855 The following standard values are defined (in hexadecimal) as *powers of*
1856 *two*, since multiple values MAY be used at the same time. For ease of
1857 understanding, the JmJobStateReasons1TC reasons are presented in the
1858 order in which the reasons are likely to occur (if implemented),
1859 starting with the 'jobIncoming' value and ending with the
1860 'jobCompletedWithErrors' value.

1861
1862 other 0x1
1863 The job state reason is not one of the standardized or
1864 registered reasons.
1865
1866 unknown 0x2
1867 The job state reason is not known to the agent or is
1868 indeterminent.
1869
1870 jobIncoming 0x4
1871 The job has been accepted by the server or device, but the
1872 server or device is expecting (1) additional operations
1873 from the client to finish creating the job and/or (2) is
1874 accessing/accepting document data.
1875
1876 submissionInterrupted 0x8
1877 The job was not completely submitted for some unforeseen
1878 reason, such as: (1) the server has crashed before the job
1879 was closed by the client, (2) the server or the document
1880 transfer method has crashed in some non-recoverable way
1881 before the document data was entirely transferred to the
1882 server, (3) the client crashed or failed to close the job
1883 before the time-out period.
1884
1885 jobOutgoing 0x10
1886 Configuration 2 only: The server is transmitting the job
1887 to the device.
1888
1889 jobHoldSpecified 0x20
1890 The value of the job's jobHold(52) attribute is TRUE. The
1891 job SHALL NOT be a candidate for processing until this
1892 reason is removed and there are no other reasons to hold
1893 the job.
1894
1895 jobHoldUntilSpecified 0x40
1896 The value of the job's jobHoldUntil(53) attribute specifies
1897 a time period that is still in the future. The job SHALL
1898 NOT be a candidate for processing until this reason is
1899 removed and there are no other reasons to hold the job.
1900

1901 jobProcessAfterSpecified 0x80
1902 The value of the job's jobProcessAfterDateAndTime(51)
1903 attribute specifies a time that is still in the future.
1904 The job SHALL NOT be a candidate for processing until this
1905 reason is removed and there are no other reasons to hold
1906 the job.
1907
1908 resourcesAreNotReady 0x100
1909 At least one of the resources needed by the job, such as
1910 media, fonts, resource objects, etc., is not ready on any
1911 of the physical devices for which the job is a candidate.
1912 This condition MAY be detected when the job is accepted, or
1913 subsequently while the job is pending or processing,
1914 depending on implementation.
1915
1916 deviceStoppedPartly 0x200
1917 One or more, but not all, of the devices to which the job
1918 is assigned are stopped. If all of the devices are stopped
1919 (or the only device is stopped), the deviceStopped reason
1920 SHALL be used.
1921
1922 deviceStopped 0x400
1923 The device(s) to which the job is assigned is (are all)
1924 stopped.
1925
1926 jobInterpreting 0x800
1927 The device to which the job is assigned is interpreting the
1928 document data.
1929
1930 jobPrinting 0x1000
1931 The output device to which the job is assigned is marking
1932 media. This value is useful for servers and output devices
1933 which spend a great deal of time processing (1) when no
1934 marking is happening and then want to show that marking is
1935 now happening or (2) when the job is in the process of
1936 being canceled or aborted while the job remains in the
1937 processing state, but the marking has not yet stopped so
1938 that impression or sheet counts are still increasing for
1939 the job.
1940
1941 jobCanceledByUser 0x2000
1942 The job was canceled by the owner of the job, i.e., by a
1943 user whose name is the same as the value of the job's
1944 jmJobOwner object, or by some other authorized end-user,
1945 such as a member of the job owner's security group.
1946
1947 jobCanceledByOperator 0x4000
1948 The job was canceled by the operator, i.e., by a user who
1949 has been authenticated as having operator privileges
1950 (whether local or remote).
1951

1952 jobCanceledAtDevice 0x8000
1953 The job was canceled by an unidentified local user, i.e., a
1954 user at a console at the device.
1955
1956 abortedBySystem 0x10000
1957 The job (1) is in the process of being aborted, (2) has
1958 been aborted by the system and placed in the 'aborted'
1959 state, or (3) has been aborted by the system and placed in
1960 the 'pendingHeld' state, so that a user or operator can
1961 manually try the job again.
1962
1963 processingToStopPoint 0x20000
1964 The requester has issued an operation to cancel or
1965 interrupt the job or the server/device has aborted the job,
1966 but the server/device is still performing some actions on
1967 the job until a specified stop point occurs or job
1968 termination/cleanup is completed.
1969
1970 This reason is recommended to be used in conjunction with
1971 the processing job state to indicate that the server/device
1972 is still performing some actions on the job while the job
1973 remains in the processing state. After all the job's
1974 resources consumed counters have stopped incrementing, the
1975 server/device moves the job from the processing state to
1976 the canceled or aborted job states.
1977
1978 serviceOffLine 0x40000
1979 The service or document transform is off-line and accepting
1980 no jobs. All pending jobs are put into the pendingHeld
1981 state. This situation could be true if the service's or
1982 document transform's input is impaired or broken.
1983
1984 jobCompletedSuccessfully 0x80000
1985 The job completed successfully.
1986
1987 jobCompletedWithWarnings 0x100000
1988 The job completed with warnings.
1989
1990 jobCompletedWithErrors 0x200000
1991 The job completed with errors (and possibly warnings too).
1992

2029

2030 **3.3.9.2 JmJobStateReasons2TC specification**

2031 The following standard values are defined (in hexadecimal) as *powers of*
2032 *two*, since multiple values MAY be used at the same time.

2033

2034 cascaded 0x1

2035 An outbound gateway has transmitted all of the job's job
2036 and document attributes and data to another spooling
2037 system.

2038

2039 deletedByAdministrator 0x2

2040 The administrator has deleted the job.

2041

2042 discardTimeArrived 0x4

2043 The job has been deleted due to the fact that the time
2044 specified by the job's job-discard-time attribute has
2045 arrived.

2046

2047 postProcessingFailed 0x8

2048 The post-processing agent failed while trying to log
2049 accounting attributes for the job; therefore the job has
2050 been placed into the completed state with the jobRetained
2051 jmJobStateReasons1 object value for a system-defined period
2052 of time, so the administrator can examine it, resubmit it,
2053 etc.

2054

2055 jobTransforming 0x10

2056 The server/device is interpreting document data and
2057 producing another electronic representation.

2058

2059 maxJobFaultCountExceeded 0x20

2060 The job has faulted several times and has exceeded the
2061 administratively defined fault count limit.

2062

2063 devicesNeedAttentionTimeOut 0x40

2064 One or more document transforms that the job is using needs
2065 human intervention in order for the job to make progress,
2066 but the human intervention did not occur within the site-
2067 settable time-out value.

2068

2069 needsKeyOperatorTimeOut 0x80

2070 One or more devices or document transforms that the job is
2071 using need a specially trained operator (who may need a key
2072 to unlock the device and gain access) in order for the job
2073 to make progress, but the key operator intervention did not
2074 occur within the site-settable time-out value.

2075

2076 jobStartWaitTimeOut 0x100
2077 The server/device has stopped the job at the beginning of
2078 processing to await human action, such as installing a
2079 special cartridge or special non-standard media, but the
2080 job was not resumed within the site-settable time-out value
2081 and the server/device has transitioned the job to the
2082 pendingHeld state.
2083

2084 jobEndWaitTimeOut 0x200
2085 The server/device has stopped the job at the end of
2086 processing to await human action, such as removing a
2087 special cartridge or restoring standard media, but the job
2088 was not resumed within the site-settable time-out value and
2089 the server/device has transitioned the job to the completed
2090 state.
2091

2092 jobPasswordWaitTimeOut 0x400
2093 The server/device has stopped the job at the beginning of
2094 processing to await input of the job's password, but the
2095 password was not received within the site-settable time-out
2096 value.
2097

2098 deviceTimedOut 0x800
2099 A device that the job was using has not responded in a
2100 period specified by the device's site-settable attribute.
2101

2102 connectingToDeviceTimeOut 0x1000
2103 The server is attempting to connect to one or more devices
2104 which may be dial-up, polled, or queued, and so may be busy
2105 with traffic from other systems, but server was unable to
2106 connect to the device within the site-settable time-out
2107 value.
2108

2109 transferring 0x2000
2110 The job is being transferred to a down stream server or
2111 downstream device.
2112

2113 queuedInDevice 0x4000
2114 The server/device has queued the job in a down stream
2115 server or downstream device.
2116

2117 jobQueued 0x8000
2118 The server/device has queued the document data.
2119

2120 jobCleanup 0x10000
2121 The server/device is performing cleanup activity as part of
2122 ending normal processing.
2123

2124 jobPasswordWait 0x20000
2125 The server/device has selected the job to be next to
2126 process, but instead of assigning resources and starting
2127 the job processing, the server/device has transitioned the
2128 job to the pendingHeld state to await entry of a password
2129 (and dispatched another job, if there is one).
2130
2131 validating 0x40000
2132 The server/device is validating the job *after* accepting the
2133 job.
2134
2135 queueHeld 0x80000
2136 The operator has held the entire job set or queue.
2137
2138 jobProofWait 0x100000
2139 The job has produced a single proof copy and is in the
2140 pendingHeld state waiting for the requester to issue an
2141 operation to release the job to print normally, obeying any
2142 job and document copy attributes that were originally
2143 submitted.
2144
2145 heldForDiagnostics 0x200000
2146 The system is running intrusive diagnostics, so that all
2147 jobs are being held.
2148
2149 noSpaceOnServer 0x800000
2150 There is no room on the server to store all of the job.
2151
2152 pinRequired 0x1000000
2153 The System Administrator settable device policy is (1) to
2154 require PINs, and (2) to hold jobs that do not have a pin
2155 supplied as an input parameter when the job was created.
2156
2157 exceededAccountLimit 0x2000000
2158 The account for which this job is drawn has exceeded its
2159 limit. This condition SHOULD be detected before the job is
2160 scheduled so that the user does not wait until his/her job
2161 is scheduled only to find that the account is overdrawn.
2162 This condition MAY also occur while the job is processing
2163 either as processing begins or part way through processing.
2164
2165 heldForRetry 0x4000000
2166 The job encountered some errors that the server/device
2167 could not recover from with its normal retry procedures,
2168 but the error might not be encountered if the job is
2169 processed again in the future. Example cases are phone
2170 number busy or remote file system in-accessible. For such
2171 a situation, the server/device SHALL transition the job
2172 from the processing to the pendingHeld, rather than to the
2173 aborted state.
2174

2175 The following values are from the X/Open PSIS draft standard:
2176
2177 canceledByShutdown 0x8000000
2178 The job was canceled because the server or device was
2179 shutdown before completing the job.
2180
2181 deviceUnavailable 0x10000000
2182 This job was aborted by the system because the device is
2183 currently unable to accept jobs.
2184
2185 wrongDevice 0x20000000
2186 This job was aborted by the system because the device is
2187 unable to handle this particular job; the spooler SHOULD
2188 try another device or the user should submit the job to
2189 another device.
2190
2191 badJob 0x40000000
2192 This job was aborted by the system because this job has a
2193 major problem, such as an ill-formed PDL; the spooler
2194 SHOULD not even try another device.
2195

2196 These bit definitions are the equivalent of a type 2 enum except that
2197 combinations of them may be used together. See section 3.7.1.2.

2198 **3.3.9.3 JmJobStateReasons3TC specification**

2199 This textual-convention is used with the jobStateReasons3 attribute to
2200 provides additional information regarding the jmJobState object. The
2201 following standard values are defined (in hexadecimal) as *powers of*
2202 *two*, since multiple values may be used at the same time:
2203

2204 jobInterruptedByDeviceFailure 0x1
2205 A device or the print system software that the job was
2206 using has failed while the job was processing. The server
2207 or device is keeping the job in the pendingHeld state until
2208 an operator can determine what to do with the job.

2209 These bit definitions are the equivalent of a type 2 enum except that
2210 combinations of them may be used together. See section 3.7.1.2. The
2211 remaining bits are reserved for future standardization and/or
2212 registration.

2213

2214 **3.3.9.4 JmJobStateReasons4TC specification**

2215 This textual-convention is used with the jobStateReasons4 attribute to
2216 provides additional information regarding the jmJobState object. The
2217 following standard values are defined (in hexadecimal) as *powers of*
2218 *two*, since multiple values MAY be used at the same time.

2219

2220

None defined at this time.

2221 These bit definitions are the equivalent of a type 2 enum except that
2222 combinations of them may be used together. See section 3.7.1.2. The
2223 remaining bits are reserved for future standardization and/or
2224 registration.

2225 **3.4 Monitoring Job Progress**

2226 There are a number of objects and attributes for monitoring the
2227 progress of a job. These objects and attributes count the number of K
2228 octets, impressions, sheets, and pages requested or completed. For
2229 impressions and sheets, "completed" means stacked, unless the
2230 implementation is unable to detect when each sheet is stacked, in which
2231 case stacked is approximated when processing of each sheet completes.
2232 There are objects and attributes for the overall job and for the
2233 current copy of the document currently being stacked. For the latter,
2234 the rate at which the various objects and attributes count depends on
2235 the sheet and document collation of the job.

2236 Job Collation included sheet collation and document collation. Sheet
2237 collation is defined to be the ordering of sheets within a document
2238 copy. Document collation is defined to be ordering of document copies
2239 within a multi-document job. There are three types of job collation
2240 (see terminology definitions in Section 2):

2241 1. uncollatedSheets(3) - No collation of the sheets within each
2242 document copy, i.e., each sheet of a document that is to
2243 produce multiple copies is replicated before the next sheet in
2244 the document is processed and stacked. If the device has an
2245 output bin collator, the uncollatedSheets(3) value may actually
2246 produce collated sheets as far as the user is concerned (in the
2247 output bins). However, when the job collation is the
2248 'uncollatedSheets(3)' value, job progress is indistinguishable
2249 to a monitoring application between a device that has an output
2250 bin collator and one that does not.

2251 2. collatedDocuments(4) - Collation of the sheets within each
2252 document copy is performed within the printing device by making
2253 multiple passes over either the source or an intermediate
2254 representation of the document. In addition, when there are
2255 multiple documents per job, the i'th copy of each document is
2256 stacked before the j'th copy of each document, i.e., the
2257 documents are collated within each job copy. For example, if a
2258 job is submitted with documents, A and B, the job is made
2259 available to the end user as: A, B, A, B, The
2260 'collatedDocuments(4)' value corresponds to the IPP [ipp-model]
2261 'separate-documents-collated-copies' value of the "multiple-
2262 document-handling" attribute.
2263

2264 If jobCopiesRequested or documentCopiesRequested = 1, then
2265 jobCollationType is defined as 4.

2266 3. uncollatedDocuments(5) - Collation of the sheets within each
2267 document copy is performed within the printing device by making
2268 multiple passes over either the source or an intermediate
2269 representation of the document. In addition, when there are
2270 multiple documents per job, all copies of the first document in
2271 the job are stacked before the any copied of the next document
2272 in the job, i.e., the documents are uncollated within the job.
2273 For example, if a job is submitted with documents, A and B, the
2274 job is mad available to the end user as: A, A, ..., B, B,
2275 The 'uncollatedDocuments(5)' value corresponds to the IPP [ipp-
2276 model] 'separate-documents-uncollated-copies' value of the
2277 "multiple-document-handling" attribute.

2278 Consider the following four variables that are used to monitor the
2279 progress of a job's impressions:

- 2280 1. jmJobImpressionsCompleted - counts the total number of
2281 impressions stacked for the job
- 2282 2. impressionsCompletedCurrentCopy - counts the number of
2283 impressions stacked for the current document copy
- 2284 3. sheetCompletedCopyNumber - identifies the number of the copy
2285 for the current document being stacked where the first copy is
2286 1.
- 2287 4. sheetCompletedDocumentNumber - identifies the current document
2288 within the job that is being stacked where the first document
2289 in a job is 1. NOTE: this attribute SHOULD NOT be implemented
2290 for implementations that only support one document per job.

2291 For each of the three types of job collation, a job with three copies
2292 of two documents (1, 2), where each document consists of 3 impressions,
2293 the four variables have the following values as each sheet is stacked
2294 for one-sided printing:

2295

2296

Job Collation Type = uncollatedSheets(3)

2297

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

2298

2299

2300

Job Collation Type = collatedDocuments(4)

2301

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

2302

2303

2304 Job Collation Type = uncollatedDocuments(5)

2305

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

2306

2307 **3.5 Job Identification**

2308 There are a number of attributes that permit a user, operator or system
 2309 administrator to identify jobs of interest, such as jobURI, jobName,
 2310 jobOriginatingHost, etc. In addition, there is a jmJobSubmissionID
 2311 object that is a text string table index. Being a table index allows a
 2312 monitoring application to quickly locate and identify a particular job
 2313 of interest that was submitted from a particular client by the user
 2314 invoking the monitoring application without having to scan the entire
 2315 job table. The Job Monitoring MIB needs to provide for identification
 2316 of the job at both sides of the job submission process. The primary
 2317 identification point is the client side. The jmJobSubmissionID allows
 2318 the monitoring application to identify the job of interest from all the
 2319 jobs currently "known" by the server or device. The value of
 2320 jmJobSubmissionID can be assigned by either the client's local system
 2321 or a downstream server or device. The point of assignment depends on
 2322 the job submission protocol in use.

2323 The server/device-side identifier, called the jmJobIndex object, SHALL
 2324 be assigned by the SNMP Job Monitoring MIB agent when the server or
 2325 device accepts the jobs from submitting clients. The jmJobIndex object
 2326 allows the interested party to obtain all objects desired that relate

2327 to a particular job. See Section 3.2, entitled 'The Job Tables and the
2328 Oldest Active and Newest Active Indexes' for the specification of how
2329 the agent SHALL assign the jmJobIndex values.

2330 The MIB provides a mapping table that maps each jmJobSubmissionID value
2331 to a corresponding jmJobIndex value generated by the agent, so that an
2332 application can determine the correct value for the jmJobIndex value
2333 for the job of interest in a single Get operation, given the Job
2334 Submission ID. See the jmJobIDGroup.

2335 In some configurations there may be more than one application program
2336 that monitors the same job when the job passes from one network entity
2337 to another when it is submitted. See configuration 3. When there are
2338 multiple job submission IDs, each entity MAY supply an appropriate
2339 jmJobSubmissionID value. In this case there would be a separate entry
2340 in the jmJobSubmissionID table, one for each jmJobSubmissionID. All
2341 entries would map to the same jmJobIndex that contains the job data.
2342 When the job is deleted, it is up to the agent to remove all entries
2343 that point to the job from the jmJobSubmissionID table as well.

2344 The jobName attribute provides a name that the user supplies as a job
2345 attribute with the job. The jobName attribute is not necessarily
2346 unique, even for one user, let alone across users.

2347 **3.5.1 The Job Submission ID specifications**

2348 This section specifies the formats for each of the registered Job
2349 Submission Ids. This format is used by the JmJobSubmissionIDTypeTC.
2350 Each job submission ID is a fixed-length, 48-octet printable US-ASCII
2351 [US-ASCII] coded character string containing no control characters,
2352 consisting of the following fields:

2353
2354 octet 1: The format letter identifying the format. The US-
2355 ASCII characters '0-9', 'A-Z', and 'a-z' are assigned in
2356 order giving 62 possible formats.
2357 octets 2-40: A 39-character, US-ASCII trailing SPACE filled
2358 field specified by the format letter, if the data is less
2359 than 39 ASCII characters.
2360 octets 41-48: A sequential or random US-ASCII number to make
2361 the ID quasi-unique.
2362

2363 If the client does not supply a job submission ID in the job submission
2364 protocol, then the agent SHALL assign a job submission ID using any of
2365 the standard formats that are reserved for the agent. Clients SHALL
2366 not use formats that are reserved for agents and agents SHALL NOT use
2367 formats that are reserved for clients, in order to reduce conflicts in
2368 ID generation. See the description for which formats are reserved for
2369 clients or for agents.

2370 Registration of additional formats may be done following the procedures
2371 described in Section 3.7.3.

2372 The format values defined at the time of completion of this
2373 specification are:

2374
2375 Format
2376 Letter Description
2377 -----
2378 '0' Job Owner generated by the server/device
2379 octets 2-40: The last 39 bytes of the jmJobOwner object.
2380 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2381 assigned by the agent.
2382 This format is reserved for agents.
2383
2384 NOTE - Clients wishing to use a job submission ID that
2385 incorporates the job owner, SHALL use format '8', not
2386 format '0'.
2387
2388 '1' Job Name
2389 octets 2-40: The last 39 bytes of the jobName attribute.
2390 octets 41-48: The US-ASCII 8-decimal-digit random number
2391 assigned by the client.
2392 This format is reserved for clients.
2393
2394 '2' Client MAC address
2395 octets 2-40: The client MAC address: in hexadecimal with each
2396 nibble of the 6 octet address being '0'-'9' or 'A' - 'F'
2397 (uppercase only). Most significant octet first.
2398 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2399 assigned by the client.
2400 This format is reserved for clients.
2401
2402 '3' Client URL
2403 octets 2-40: The last 39 bytes of the client URL [URI-spec].
2404 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2405 assigned by the client.
2406 This format is reserved for clients.
2407
2408 '4' Job URI
2409 octets 2-40: The last 39 bytes of the URI [URI-spec] assigned
2410 by the server or device to the job when the job was
2411 submitted for processing.
2412 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2413 assigned by the agent.
2414 This format is reserved for agents.
2415
2416 '5' POSIX User Number
2417 octets 2-40: The last 39 bytes of a user number, such as POSIX
2418 user number.
2419 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2420 assigned by the client.

2421 This format is reserved for clients.
2422
2423 '6' User Account Number
2424 octets 2-40: The last 39 bytes of the user account number.
2425 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2426 assigned by the client.
2427 This format is reserved for clients.
2428
2429 '7' DTMF Incoming FAX routing number
2430 octets 2-40: The last 39 bytes of the DTMF incoming FAX
2431 routing number.
2432 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2433 assigned by the client.
2434 This format is reserved for clients.
2435
2436 '8' Job Owner supplied by the client
2437 octets 2-40: The last 39 bytes of the job owner name (that the
2438 agent returns in the jmJobOwner object).
2439 octets 41-48: The US-ASCII 8-decimal-digit sequential number
2440 assigned by the client.
2441 This format is reserved for clients. See format '0' which is
2442 reserved for agents.
2443
2444 '9' Host Name
2445 octets 2-40: The last 39 bytes of the host name with trailing
2446 SPACES that submitted the job to this server/device using a
2447 protocol, such as LPD [RFC1179] which includes the host
2448 name in the job submission protocol.
2449 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2450 representation of the job id generated by the submitting
2451 server (configuration 3) or the client (configuration 1 and
2452 2), such as in the LPD protocol.
2453 This format is reserved for clients.
2454
2455 'A' AppleTalk Protocol
2456 octets 2-40: Contains the AppleTalk printer name, with the
2457 first character of the name in octet 2. AppleTalk printer
2458 names are a maximum of 31 characters. Any unused portion
2459 of this field shall be filled with spaces.
2460 octets 41-48: '00000XXX', where 'XXX' is the 3-digit US-ASCII
2461 decimal representation of the Connection Id.
2462 This format is reserved for agents.
2463

2464 'B' NetWare PServer
2465 octets 2-40: Contains the Directory Path Name as recorded by
2466 the Novell File Server in the queue directory. If the
2467 string is less than 40 octets, the left-most character in
2468 the string shall appear in octet position 2. Otherwise,
2469 only the last 39 bytes shall be included. Any unused
2470 portion of this field shall be filled with spaces.
2471 octets 41-48: '000XXXXX' The US-ASCII representation of the
2472 Job Number as per the NetWare File Server Queue Management
2473 Services.
2474 This format is reserved for agents.
2475
2476 'C' Server Message Block protocol (SMB)
2477 octets 2-40: Contains a decimal (US-ASCII coded)
2478 representation of the 16 bit SMB Tree Id field, which
2479 uniquely identifies the connection that submitted the job
2480 to the printer. The most significant digit of the numeric
2481 string shall be placed in octet position 2. All unused
2482 portions of this field shall be filled with spaces. The
2483 SMB Tree Id has a maximum value of 65,535.
2484 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2485 representation of the File Handle returned from the device
2486 to the client in response to a Create Print File command.
2487 This format is reserved for agents.
2488
2489 'D' Transport Independent Printer/System Interface (TIP/SI)
2490 octets 2-40: Contains the Job Name from the Job Control-Start
2491 Job (JC-SJ) command. If the Job Name portion is less than
2492 40 octets, the left-most character in the string shall
2493 appear in octet position 2. Any unused portion of this
2494 field shall be filled with spaces. Otherwise, only the
2495 last 39 bytes shall be included.
2496 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2497 representation of the jmJobIndex assigned by the agent.
2498 This format is reserved for agents, since the agent supplies
2499 octets 41-48, though the client supplies the job name. See
2500 format '1' reserved to clients to submit job name ids in
2501 which they supply octets 41-48.
2502
2503 'E' IPDS on the MVS or VSE platform
2504
2505 octets 2-40: Contains bytes 2-27 of the XOH Define Group
2506 Boundary Group ID triplet. Octet position 2 MUST carry the
2507 value x'01'. Bytes 28-40 MUST be filled with spaces.
2508 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2509 representation of the jmJobIndex assigned by the agent.
2510 This format is reserved for agents, since the agent supplies
2511 octets 41-48, though the client supplies the job name.
2512

2513 'F' IPDS on the VM platform
2514 octets 2-40: Contains bytes 2-31 of the XOH Define Group
2515 Boundary Group ID triplet. Octet position 2 MUST carry the
2516 value x'02'. Bytes 32-40 MUST be filled with spaces.
2517 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2518 representation of the jmJobIndex assigned by the agent.
2519 This format is reserved for agents, since the agent supplies
2520 octets 41-48, though the client supplies the file name.
2521
2522 'G' IPDS on the OS/400 platform
2523 octets 2-40: Contains bytes 2-36 of the XOH Define Group
2524 Boundary Group ID triplet. Octet position 2 MUST carry the
2525 value x'03'. Bytes 37-40 MUST be filled with spaces.
2526 octets 41-48: The US-ASCII 8-decimal-digit leading zero
2527 representation of the jmJobIndex assigned by the agent.
2528 This format is reserved for agents, since the agent supplies
2529 octets 41-48, though the client supplies the job name.
2530

2531 NOTE - the job submission id is only intended to be unique between a
2532 limited set of clients for a limited duration of time, namely, for the
2533 life time of the job in the context of the server or device that is
2534 processing the job. Some of the formats include something that is
2535 unique per client and a random number so that the same job submitted by
2536 the same client will have a different job submission id. For other
2537 formats, where part of the id is guaranteed to be unique for each
2538 client, such as the MAC address or URL, a sequential number SHOULD
2539 suffice for each client (and may be easier for each client to manage).
2540 Therefore, the length of the job submission id has been selected to
2541 reduce the probability of collision to an extremely low number, but is
2542 not intended to be an absolute guarantee of uniqueness. None-the-less,
2543 collisions are remotely possible, but without bad consequences, since
2544 this MIB is intended to be used only for monitoring jobs, not for
2545 controlling and managing them.

2546

2547

2548 **3.6 Internationalization Considerations**

2549 This section describes the internationalization considerations included
2550 in this MIB.

2551 3.6.1 Text generated by the server or device

2552 There are a few objects and attributes generated by the server or
2553 device that SHALL be represented using the Universal Multiple-Octet
2554 Coded Character Set (UCS) [ISO-10646]. These objects and attributes
2555 are always supplied (if implemented) by the agent, not by the job
2556 submitting client:

- 2557 1. jmGeneralJobSetName object
- 2558 2. processingMessage(6) attribute
- 2559 3. physicalDevice(32) (name value) attribute

2560 The character encoding scheme for representing these objects and
2561 attributes SHALL be UTF-8 as REQUIRED by RFC 2277 [RFC2277]. The
2562 'JmUTF8StringTC' textual convention is used to indicate UTF-8 text
2563 strings.

2564 NOTE - For strings in 7-bit US-ASCII, there is no impact since the UTF-
2565 8 representation of 7-bit ASCII is identical to the US-ASCII [US-ASCII]
2566 encoding.

2567 The text contained in the processingMessage(6) attribute is generated
2568 by the server/device. The natural language for the
2569 processingMessage(6) attribute is identified by the
2570 processingMessageNaturalLangTag(7) attribute. The
2571 processingMessageNaturalLangTag(7) attribute uses the
2572 JmNaturalLanguageTagTC textual convention which SHALL conform to the
2573 language tag mechanism specified in RFC 1766 [RFC-1766]. The
2574 JmNaturalLanguageTagTC value is the same as the IPP [IPP-model]
2575 'naturalLanguage' attribute syntax. RFC 1766 specifies that a US-ASCII
2576 string consisting of the natural language followed by an optional
2577 country field. Both fields use the same two-character codes from ISO
2578 639 [ISO-639] and ISO 3166 [ISO-3166], respectively, that are used in
2579 the Printer MIB for identifying language and country.

2580 Examples of the values of the processingMessageNaturalLangTag(7)
2581 attribute include:

- 2582 1. 'en' for English
- 2583 2. 'en-us' for US English
- 2584 3. 'fr' for French
- 2585 4. 'de' for German

2586

2587 3.6.2 Text supplied by the job submitter

2588 All of the objects and attributes represented by the 'JmJobStringTC'
2589 textual-convention are either (1) supplied in the job submission
2590 protocol by the client that submits the job to the server or device or
2591 (2) are defaulted by the server or device if the job submitting client
2592 does not supply values. The agent SHALL represent these objects and
2593 attributes in the MIB either (1) in the coded character set as they
2594 were submitted or (2) MAY convert the coded character set to another
2595 coded character set or encoding scheme. In any case, the resulting
2596 coded character set representation SHOULD be UTF-8 [UTF-8], but SHALL
2597 be one in which the code positions from 0 to 31 is not used, 32 to 127
2598 is US-ASCII [US-ASCII], 127 is not unused, and the remaining code
2599 positions 128 to 255 represent single-byte or multi-byte graphic
2600 characters structured according to ISO 2022 [ISO-2022] or are unused.

2601 The coded character set SHALL be one of the ones registered with IANA
2602 [IANA] and SHALL be identified by the jobCodedCharSet attribute in the
2603 jmJobAttributeTable for the job. If the agent does not know what coded
2604 character set was used by the job submitting client, the agent SHALL
2605 either (1) return the 'unknown(2)' value for the jobCodedCharSet
2606 attribute or (2) not return the jobCodedCharSet attribute for the job.

2607 Examples of coded character sets which meet this criteria for use as
2608 the value of the jobCodedCharSet job attribute are: US-ASCII [US-
2609 ASCII], ISO 8859-1 (Latin-1) [ISO-8859-1], any ISO 8859-n, HP Roman8,
2610 IBM Code Page 850, Windows Default 8-bit set, UTF-8 [UTF-8], US-ASCII
2611 plus JIS X0208-1990 Japanese [JIS X0208], US-ASCII plus GB2312-1980 PRC
2612 Chinese [GB2312]. See the IANA registry of coded character sets [IANA
2613 charsets].

2614 Examples of coded character sets which do not meet this criteria are:
2615 national 7-bit sets conforming to ISO 646 (except US-ASCII), EBCDIC,
2616 and ISO 10646 (Unicode) [ISO-10646]. In order to represent Unicode
2617 characters, the UTF-8 [UTF-8] encoding scheme SHALL be used which has
2618 been assigned the MIBenum value of '106' by IANA.

2619 The jobCodedCharSet attribute uses the imported 'CodedCharSet' textual-
2620 convention from the Printer MIB [printmib].

2621 The natural language for attributes represented by the textual-
2622 convention JmJobStringTC is identified either (1) by the
2623 jobNaturalLanguageTag(9) attribute or is keywords in US-English (as in
2624 IPP). A monitoring application SHOULD attempt to localize keywords
2625 into the language of the user by means of some lookup mechanism. If
2626 the keyword value is not known to the monitoring application, the
2627 monitoring application SHOULD assume that the value is in the natural
2628 language specified by the job's jobNaturalLanguageTag(9) attribute and
2629 SHOULD present the value to its user as is. The

2630 jobNaturalLanguageTag(9) attribute value SHALL have the same syntax and
2631 semantics as the processingMessageNaturalLangTag(7) attribute, except
2632 that the jobNaturalLanguageTag(9) attribute identifies the natural
2633 language of attributes supplied by the job submitter instead of the
2634 natural language of the processingMessage(6) attribute. See Section
2635 3.6.1.

2636 3.6.3 'DateAndTime' for representing the date and time

2637 This MIB also contains objects that are represented using the
2638 DateAndTime textual convention from SMIV2 [SMIV2-TC]. The job
2639 management application SHALL display such objects in the locale of the
2640 user running the monitoring application.

2641 3.7 IANA and PWG Registration Considerations

2642 This MIB does not require any additional registration schemes for IANA,
2643 but does depend on registration schemes that other Internet standards
2644 track specifications have set up. The names of these IANA registration
2645 assignments under the /in-notes/iana/assignments/ path:

- 2646 1. printer-language-numbers - used as enums in the documentFormat(38)
2647 attribute
- 2648 2. media-types - uses as keywords in the documentFormat(38) attribute
- 2649 3. character-sets - used as enums in the jobCodedCharSet(8) attribute

2650 The Printer Working Group (PWG) will handle registration of additional
2651 enums after approving this standard, according to the procedures
2652 described in this section:

2653 3.7.1 PWG Registration of enums

2654 This specification uses textual conventions to define enumerated values
2655 (enums) and bit values. Enumerations (enums) and bit values are sets
2656 of symbolic values defined for use with one or more objects or
2657 attributes. All enumeration sets and bit value sets are assigned a
2658 symbolic data type name (textual convention). As a convention the
2659 symbolic name ends in "TC" for textual convention. These enumerations
2660 are defined at the beginning of the MIB module specification.

2661 The PWG has defined several type of enumerations for use in the Job
2662 Monitoring MIB and the Printer MIB[print-mib]. These types differ in
2663 the method employed to control the addition of new enumerations.
2664 Throughout this document, references to "type n enum", where n can be
2665 1, 2 or 3 can be found in the various tables. The definitions of these
2666 types of enumerations are:

2667 3.7.1.1 Type 1 enumerations

2668 Type 1 enumeration: All the values are defined in the Job Monitoring
2669 MIB specification (RFC for the Job Monitoring MIB). Additional
2670 enumerated values require a new RFC.

2671 There are no type 1 enums in the current draft.

2672 3.7.1.2 Type 2 enumerations

2673 Type 2 enumeration: An initial set of values are defined in the Job
2674 Monitoring MIB specification. Additional enumerated values are
2675 registered with the PWG.

2676 The following type 2 enums are contained in the current draft :

- 2677 1. JmUTF8StringTC
- 2678 2. JmJobStringTC
- 2679 3. JmNaturalLanguageTagTC
- 2680 4. JmTimeStampTC
- 2681 5. JmFinishingTC [same enum values as IPP "finishing" attribute]
- 2682 6. JmPrintQualityTC [same enum values as IPP "print-quality"
- 2683 attribute]
- 2684 7. JmTonerEconomyTC
- 2685 8. JmMediumTypeTC
- 2686 9. JmJobSubmissionIDTypeTC
- 2687 10. JmJobCollationTypeTC
- 2688 11. JmJobStateTC [same enum values as IPP "job-state" attribute]
- 2689 12. JmAttributeTypeTC

2690 For those textual conventions that have the same enum values as the
2691 indicated IPP Job attribute are simultaneously registered by the PWG
2692 for use with IPP [ipp-model] and the Job Monitoring MIB.

2693 3.7.1.3 Type 3 enumeration

2694 Type 3 enumeration: An initial set of values are defined in the Job
2695 Monitoring MIB specification. Additional enumerated values are
2696 registered through the PWG without PWG review.

2697 There are no type 3 enums in the current draft.

2698

2699 3.7.2 PWG Registration of type 2 bit values

2700 This draft contains the following type 2 bit value textual-conventions:
2701 1. JmJobServiceTypesTC
2702 2. JmJobStateReasons1TC
2703 3. JmJobStateReasons2TC
2704 4. JmJobStateReasons3TC
2705 5. JmJobStateReasons4TC

2706 These textual-conventions are defined as bits in an Integer so that
2707 they can be used with SNMPv1 SMI. The jobStateReasonsN (N=1..4)
2708 attributes are defined as bit values using the corresponding
2709 JmJobStateReasonsMTC textual-conventions.

2710 The registration of JmJobServiceTypesTC and JmJobStateReasonsMTC bit
2711 values follow the procedures for a type 2 enum as specified in Section
2712 3.7.1.2.

2713 3.7.3 PWG Registration of Job Submission Id Formats

2714 In addition to enums and bit values, this specification assigns a
2715 single ASCII digit or letter to various job submission ID formats. See
2716 the JmJobSubmissionIDTypeTC textual-convention and the object. The
2717 registration of JobSubmissionID format numbers follows the procedures
2718 for a type 2 enum as specified in Section 3.7.1.2.

2719 3.7.4 PWG Registration of MIME types/sub-types for document-formats

2720 The documentFormat(38) attribute has MIME type/sub-type values for
2721 indicating document formats which IANA registers as "media type" names.
2722 The values of the documentFormat(38) attribute are the same as the
2723 corresponding Internet Printing Protocol (IPP) "document-format" Job
2724 attribute values [ipp-model].

2725 **3.8 Security Considerations**

2726 3.8.1 Read-Write objects

2727 All objects are read-only, greatly simplifying the security
2728 considerations. If another MIB augments this MIB, that MIB might
2729 accept SNMP Write operations to objects in that MIB whose effect is to
2730 modify the values of read-only objects in this MIB. However, that MIB
2731 SHALL have to support the required access control in order to achieve
2732 security, not this MIB.

2733 3.8.2 Read-Only Objects In Other User's Jobs

2734 The security policy of some sites MAY be that unprivileged users can
2735 only get the objects from jobs that they submitted, plus a few minimal
2736 objects from other jobs, such as the jmJobKOctetsPerCopyRequested and
2737 jmJobKOctetsProcessed objects, so that a user can tell how busy a
2738 printer is. Other sites MAY allow all unprivileged users to see all
2739 objects of all jobs. This MIB does not require, nor does it specify
2740 how, such restrictions would be implemented. A monitoring application
2741 SHOULD enforce the site security policy with respect to returning
2742 information to an unprivileged end user that is using the monitoring
2743 application to monitor jobs that do not belong to that user, i.e., the
2744 jmJobOwner object in the jmJobTable does not match the user's user
2745 name.

2746 An operator is a privileged user that would be able to see all objects
2747 of all jobs, independent of the policy for unprivileged users.

2748 **3.9 Notifications**

2749 This MIB does not specify any notifications. For simplicity,
2750 management applications are expected to poll for status. The
2751 jmGeneralJobPersistence and jmGeneralAttributePersistence objects
2752 assist an application to determine the polling rate. The resulting
2753 network traffic is not expected to be significant.

2754 4 MIB specification

2755 The following pages constitute the actual Job Monitoring MIB.

```
2756 Job-Monitoring-MIB DEFINITIONS ::= BEGIN
2757
2758 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

2759
2760 -- Use the enterprises arc assigned to the PWG which is pwg(2699).
2761 -- Group all PWG mibs under mibs(1).
2762
2763 jobmonMIB MODULE-IDENTITY
2764     LAST-UPDATED "9902200000Z"
2765     ORGANIZATION "Printer Working Group (PWG)"
2766     CONTACT-INFO
2767         "Tom Hastings
2768         Postal:  Xerox Corp.
2769                 Mail stop ESAE-231
2770                 701 S. Aviation Blvd.
2771                 El Segundo, CA 90245
2772
2773         Tel:    (301)333-6413
2774         Fax:    (301)333-5514
2775         E-mail: hastings@cpl0.es.xerox.com
2776
2777         Send questions and comments to the Printer Working Group (PWG)
2778         using the Job Monitoring Project (JMP) Mailing List:
2779         jmp@pwg.org
2780
2781         For further information, including how to subscribe to the
2782         jmp mailing list, access the PWG web page under 'JMP':
2783
2784         http://www.pwg.org/
2785
2786         Implementers of this specification are encouraged to join the
2787         jmp mailing list in order to participate in discussions on any
2788         clarifications needed and registration proposals being reviewed
2789         in order to achieve consensus."
2790     DESCRIPTION
2791         "The MIB module for monitoring job in servers, printers, and
2792         other devices.
2793
2794         Version: 2.0"
2795     ::= { enterprises pwg(2699) mibs(1) jobmonMIB(1) }
```

2796
2797
2798 -- Textual conventions for this MIB module
2799
2800 JmUTF8StringTC ::= TEXTUAL-CONVENTION
2801 DISPLAY-HINT "255a"
2802 STATUS current
2803 DESCRIPTION
2804 "To facilitate internationalization, this TC represents
2805 information taken from the ISO/IEC IS 10646-1 character set,
2806 encoded as an octet string using the UTF-8 character encoding
2807 scheme.
2808
2809 See section 3.6.1, entitled: 'Text generated by the server or
2810 device'."
2811 SYNTAX OCTET STRING (SIZE (0..63))
2812
2813
2814
2815
2816 JmJobStringTC ::= TEXTUAL-CONVENTION
2817 STATUS current
2818 DESCRIPTION
2819 "To facilitate internationalization, this TC represents
2820 information using any coded character set registered by IANA as
2821 specified in section 3.7. While it is recommended that the
2822 coded character set be UTF-8 [UTF-8], the actual coded
2823 character set SHALL be indicated by the value of the
2824 jobCodedCharSet(8) attribute for the job.
2825
2826 See section 3.6.2, entitled: 'Text supplied by the job
2827 submitter'."
2828 SYNTAX OCTET STRING (SIZE (0..63))
2829
2830
2831
2832
2833 JmNaturalLanguageTagTC ::= TEXTUAL-CONVENTION
2834 STATUS current
2835 DESCRIPTION
2836 "An IETF RFC 1766-compliant 'language tag', with zero or more
2837 sub-tags that identify a natural language. While RFC 1766
2838 specifies that the US-ASCII values are case-insensitive, this
2839 MIB specification requires that all characters SHALL be lower
2840 case in order to simplify comparing by management applications.
2841
2842 See section 3.6.1, entitled: 'Text generated by the server or
2843 device' and section 3.6.2, entitled: 'Text supplied by the job
2844 submitter'."
2845 SYNTAX OCTET STRING (SIZE (0..63))

```
2846
2847
2848 JmTimeStampTC ::= TEXTUAL-CONVENTION
2849     STATUS      current
2850     DESCRIPTION
2851         "The simple time at which an event took place.  The units are
2852         in seconds since the system was booted.
2853
2854         NOTE - JmTimeStampTC is defined in units of seconds, rather
2855         than 100ths of seconds, so as to be simpler for agents to
2856         implement (even if they have to implement the 100ths of a
2857         second to comply with implementing sysUpTime in MIB-II[mib-
2858         II].)
2859
2860         NOTE - JmTimeStampTC is defined as an Integer32 so that it can
2861         be used as a value of an attribute, i.e., as a value of the
2862         jmAttributeValueAsInteger object.  The TimeStamp textual-
2863         convention defined in SNMPv2-TC [SMIV2-TC] is defined as an
2864         APPLICATION 3 IMPLICIT INTEGER tag, not an Integer32 which is
2865         defined in SNMPv2-SMI [SMIV2-TC] as UNIVERSAL 2 IMPLICIT
2866         INTEGER, so cannot be used in this MIB as one of the values of
2867         jmAttributeValueAsInteger."
2868     SYNTAX      INTEGER (0..2147483647)
2869
2870
2871
2872
2873 JmJobSourcePlatformTypeTC ::= TEXTUAL-CONVENTION
2874     STATUS      current
2875     DESCRIPTION
2876         "The source platform type that can submit jobs to servers or
2877         devices in any of the 3 configurations.
2878
2879         This is a type 2 enumeration.  See Section 3.7.1.2.  See also
2880         IANA operating-system-names registry."
2881     SYNTAX      INTEGER {
                other(1),
                unknown(2),
                sptUNIX(3),           -- UNIX
                sptOS2(4),           -- OS/2
                sptPCDOS(5),        -- DOS
                sptNT(6),           -- NT
                sptMVS(7),          -- MVS
                sptVM(8),           -- VM
                sptOS400(9),        -- OS/400
                sptVMS(10),         -- VMS
                sptWindows(11),     -- Windows
                sptNetWare(12)      -- NetWare
2882     }
```



```
2883
2884
2885 JmFinishingTC ::= TEXTUAL-CONVENTION
2886     STATUS      current
2887     DESCRIPTION
2888         "The type of finishing operation.
2889
2890         These values are the same as the enum values of the IPP
2891         'finishings' attribute.  See Section 3.7.1.2.
2892
2893         other(1),
2894             Some other finishing operation besides one of the specified
2895             or registered values.
2896
2897         unknown(2),
2898             The finishing is unknown.
2899
2900         none(3),
2901             Perform no finishing.
2902
2903         staple(4),
2904             Bind the document(s) with one or more staples. The exact
2905             number and placement of the staples is site-defined.
2906
2907         punch(5),
2908             Holes are required in the finished document. The exact
2909             number and placement of the holes is site-defined. The
2910             punch specification MAY be satisfied (in a site- and
2911             implementation-specific manner) either by
2912             drilling/punching, or by substituting pre-drilled media.
2913
2914         cover(6),
2915             Select a non-printed (or pre-printed) cover for the
2916             document. This does not supplant the specification of a
2917             printed cover (on cover stock medium) by the document
2918             itself.
2919
2920         bind(7)
2921             Binding is to be applied to the document; the type and
2922             placement of the binding is product-specific.
2923
2924         This is a type 2 enumeration.  See Section 3.7.1.2."
2925     SYNTAX      INTEGER {
2926         other(1),
2927         unknown(2),
2928         none(3),
2929         staple(4),
2930         punch(5),
2931         cover(6),
2932         bind(7)
2933     }
```

```
2934
2935
2936 JmPrintQualityTC ::= TEXTUAL-CONVENTION
2937     STATUS          current
2938     DESCRIPTION
2939         "Print quality settings.
2940
2941         These values are the same as the enum values of the IPP 'print-
2942         quality' attribute.  See Section 3.7.1.2.
2943
2944         This is a type 2 enumeration.  See Section 3.7.1.2."
2945     SYNTAX          INTEGER {
2946         other(1),      -- Not one of the specified or registered
2947                        -- values.
2948         unknown(2),   -- The actual value is unknown.
2949         draft(3),     -- Lowest quality available on the printer.
2950         normal(4),    -- Normal or intermediate quality on the
2951                        -- printer.
2952         high(5)       -- Highest quality available on the printer.
2953     }
2954
2955 JmPrinterResolutionTC ::= TEXTUAL-CONVENTION
2956     STATUS          current
2957     DESCRIPTION
2958         "Printer resolutions.
2959
2960         Nine octets consisting of two 4-octet SIGNED-INTEGERS followed
2961         by a SIGNED-BYTE.  The values are the same as those specified
2962         in the Printer MIB [printmib].  The first SIGNED-INTEGER
2963         contains the value of prtMarkerAddressabilityXFeedDir.  The
2964         second SIGNED-INTEGER contains the value of
2965         prtMarkerAddressabilityFeedDir.  The SIGNED-BYTE contains the
2966         value of prtMarkerAddressabilityUnit.
2967
2968         Note: the latter value is either 3 (tenThousandsOfInches) or 4
2969         (micrometers) and the addressability is in 10,000 units of
2970         measure.  Thus the SIGNED-INTEGERS represent integral values in
2971         either dots-per-inch or dots-per-centimeter.
2972
2973         The syntax is the same as the IPP 'printer-resolution'
2974         attribute.  See Section 3.7.1.2."
2975     SYNTAX          OCTET STRING (SIZE(9))
```

```
2973
2974
2975 JmTonerEconomyTC ::= TEXTUAL-CONVENTION
2976     STATUS      current
2977     DESCRIPTION
2978         "Toner economy settings.
2979
2980         This is a type 2 enumeration.  See Section 3.7.1.2."
2981     SYNTAX      INTEGER {
2982         unknown(2),      -- unknown.
2983         off(3),          -- Off. Normal. Use full toner.
2984         on(4)            -- On. Use less toner than normal.
2985     }
2986
2987 JmBooleanTC ::= TEXTUAL-CONVENTION
2988     STATUS      current
2989     DESCRIPTION
2990         "Boolean true or false value.
2991
2992         This is a type 2 enumeration.  See Section 3.7.1.2."
2993     SYNTAX      INTEGER {
2994         unknown(2),      -- unknown.
2995         false(3),        -- FALSE.
2996         true(4)          -- TRUE.
2997     }
2998
2999 JmMediumTypeTC ::= TEXTUAL-CONVENTION
3000     STATUS      current
3001     DESCRIPTION
3002         "Identifies the type of medium.
3003
3004         other(1),
3005             The type is neither one of the values listed in this
3006             specification nor a registered value.
3007
3008         unknown(2),
3009             The type is not known.
3010
3011         stationery(3),
3012             Separately cut sheets of an opaque material.
3013
3014         transparency(4),
3015             Separately cut sheets of a transparent material.
3016
3017         envelope(5),
3018             Envelopes that can be used for conventional mailing
3019             purposes.
```

```
3018
3019     envelopePlain(6),
3020         Envelopes that are not preprinted and have no windows.
3021
3022     envelopeWindow(7),
3023         Envelopes that have windows for addressing purposes.
3024
3025     continuousLong(8),
3026         Continuously connected sheets of an opaque material
3027         connected along the long edge.
3028
3029     continuousShort(9),
3030         Continuously connected sheets of an opaque material
3031         connected along the short edge.
3032
3033     tabStock(10),
3034         Media with tabs.
3035
3036     multiPartForm(11),
3037         Form medium composed of multiple layers not pre-attached to
3038         one another; each sheet MAY be drawn separately from an
3039         input source.
3040
3041     labels(12),
3042         Label-stock.
3043
3044     multiLayer(13)
3045         Form medium composed of multiple layers which are pre-
3046         attached to one another, e.g. for use with impact printers.
3047
3048     This is a type 2 enumeration. See Section 3.7.1.2. These enum
3049     values correspond to the keyword name strings of the
3050     prtInputMediaType object in the Printer MIB [print-mib]. There
3051     is no printer description attribute in IPP/1.0 that represents
3052     these values."
3053     SYNTAX      INTEGER {
3054         other(1),
3055         unknown(2),
3056         stationery(3),
3057         transparency(4),
3058         envelope(5),
3059         envelopePlain(6),
3060         envelopeWindow(7),
3061         continuousLong(8),
3062         continuousShort(9),
3063         tabStock(10),
3064         multiPartForm(11),
3065         labels(12),
3066         multiLayer(13)
3067     }
3068
```

```
3069
3070
3071 JmJobCollationTypeTC ::= TEXTUAL-CONVENTION
3072     STATUS      current
3073     DESCRIPTION
3074         "This value is the type of job collation. Implementations that
3075         don't support multiple documents or don't support multiple
3076         copies SHALL NOT support the uncollatedDocuments(5) value.
3077
3078         This is a type 2 enumeration. See Section 3.7.1.2. See also
3079         Section 3.4, entitled 'Monitoring Job Progress'."
3080     SYNTAX      INTEGER {
3081         other(1),
3082         unknown(2),
3083         uncollatedSheets(3),      -- sheets within each document copy
3084                                   -- are not collated: 1 1 ..., 2 2 ...,
3085                                   -- No corresponding value of IPP
3086                                   -- "multiple-document-handling"
3087         collatedDocuments(4),    -- internal collated sheets,
3088                                   -- documents: A, B, A, B, ...
3089                                   -- Corresponds to IPP "multiple-
3090                                   -- document-handling"='separate-
3091                                   -- documents-collated-copies'
3092         uncollatedDocuments(5)  -- internal collated sheets,
3093                                   -- documents: A, A, ..., B, B, ...
3094                                   -- Corresponds to IPP "multiple-
3095                                   -- document-handling"='separate-
3096                                   -- documents-uncollated-copies'
3097     }
3098
3099
3100 JmJobSubmissionIDTypeTC ::= TEXTUAL-CONVENTION
3101     STATUS      current
3102     DESCRIPTION
3103         "Identifies the format type of a job submission ID.
3104
3105         Each job submission ID is a fixed-length, 48-octet printable
3106         US-ASCII [US-ASCII] coded character string containing no
3107         control characters, consisting of the fields defined in section
3108         3.5.1.
3109
3110         This is like a type 2 enumeration. See section 3.7.3."
3111     SYNTAX      OCTET STRING(SIZE(1)) -- ASCII '0'-'9', 'A'-'Z', 'a'-'z'
```

```

3112
3113
3114 JmJobStateTC ::= TEXTUAL-CONVENTION
3115     STATUS      current
3116     DESCRIPTION
3117         "The current state of the job (pending, processing, completed,
3118         etc.). The following figure shows the normal job state
3119         transitions:
3120
3121                                     +----> canceled(7)
3122                                     /
3123     +----> pending(3) -----> processing(5) -----+-----> completed(9)
3124     |           ^           |           ^           |
3125     |           |           |           |           |
3126     |           v           |           v           |
3127     +----> pendingHeld(4)  processingStopped(6) ----+
3128

```

3129 Figure 4 - Normal Job State Transitions

3130
3131 Normally a job progresses from left to right. Other state
3132 transitions are unlikely, but are not forbidden. Not shown are
3133 the transitions to the canceled state from the pending,
3134 pendingHeld, and processingStopped states.

3135
3136 Jobs in the pending, processing, and processingStopped states
3137 are called 'active', while jobs in the pendingHeld, canceled,
3138 aborted, and completed states are called 'inactive'. Jobs
3139 reach one of the three terminal states: completed, canceled, or
3140 aborted, *after* the jobs have completed all activity, and all
3141 MIB objects and attributes have reached their final values for
3142 the job.

3143
3144 These values are the same as the enum values of the IPP 'job-
3145 state' job attribute. See Section 3.7.1.2.

3146
3147 unknown(2),
3148 The job state is *not* known, or its state is indeterminate.

3149
3150 pending(3),
3151 The job is a candidate to start processing, but is not yet
3152 processing.

3153
3154 pendingHeld(4),
3155 The job is not a candidate for processing for any number of
3156 reasons but will return to the pending state as soon as the
3157 reasons are no longer present. The job's
3158 jmJobStateReasons1 object and/or jobStateReasonsN (N=2..4)
3159 attributes SHALL indicate why the job is no longer a
3160 candidate for processing. The reasons are represented as
3161 bits in the jmJobStateReasons1 object and/or
3162 jobStateReasonsN (N=2..4) attributes. See the

3163 JmJobStateReasonsMTC (N=1..4) textual convention for the
3164 specification of each reason.
3165
3166 processing(5),
3167 One or more of:
3168
3169 1. the job is using, or is attempting to use, one or more
3170 purely software processes that are analyzing, creating, or
3171 interpreting a PDL, etc.,
3172
3173 2. the job is using, or is attempting to use, one or more
3174 hardware devices that are interpreting a PDL, making marks
3175 on a medium, and/or performing finishing, such as stapling,
3176 etc., OR
3177
3178 3. (configuration 2) the server has made the job ready for
3179 printing, but the output device is not yet printing it,
3180 either because the job hasn't reached the output device or
3181 because the job is queued in the output device or some
3182 other spooler, awaiting the output device to print it.
3183
3184 When the job is in the processing state, the entire job
3185 state includes the detailed status represented in the
3186 device MIB indicated by the hrDeviceIndex value of the
3187 job's physicalDevice attribute, if the agent implements
3188 such a device MIB.
3189
3190 Implementations MAY, though they NEED NOT, include
3191 additional values in the job's jmJobStateReasons1 object to
3192 indicate the progress of the job, such as adding the
3193 jobPrinting value to indicate when the device is actually
3194 making marks on a medium and/or the processingToStopPoint
3195 value to indicate that the server or device is in the
3196 process of canceling or aborting the job.
3197
3198 processingStopped(6),
3199 The job has stopped while processing for any number of
3200 reasons and will return to the processing state as soon as
3201 the reasons are no longer present.
3202
3203 The job's jmJobStateReasons1 object and/or the job's
3204 jobStateReasonsN (N=2..4) attributes MAY indicate why the
3205 job has stopped processing. For example, if the output
3206 device is stopped, the deviceStopped value MAY be included
3207 in the job's jmJobStateReasons1 object.
3208
3209 NOTE - When an output device is stopped, the device usually
3210 indicates its condition in human readable form at the
3211 device. The management application can obtain more
3212 complete device status remotely by querying the appropriate
3213 device MIB using the job's deviceIndex attribute(s), if the
3214 agent implements such a device MIB

3215
3216 canceled(7),
3217 A client has canceled the job and the server or device has
3218 completed canceling the job *AND* all MIB objects and
3219 attributes have reached their final values for the job.
3220 While the server or device is canceling the job, the job's
3221 jmJobStateReasons1 object *SHOULD* contain the
3222 processingToStopPoint value and one of the canceledByUser,
3223 canceledByOperator, or canceledAtDevice values. The
3224 canceledByUser, canceledByOperator, or canceledAtDevice
3225 values remain while the job is in the canceled state.
3226
3227 aborted(8),
3228 The job has been aborted by the system, usually while the
3229 job was in the processing or processingStopped state and
3230 the server or device has completed aborting the job *AND* all
3231 MIB objects and attributes have reached their final values
3232 for the job. While the server or device is aborting the
3233 job, the job's jmJobStateReasons1 object *MAY* contain the
3234 processingToStopPoint and abortedBySystem values. If
3235 implemented, the abortedBySystem value *SHALL* remain while
3236 the job is in the aborted state.
3237
3238 completed(9)
3239 The job has completed successfully or with warnings or
3240 errors after processing and all of the media have been
3241 successfully stacked in the appropriate output bin(s) *AND*
3242 all MIB objects and attributes have reached their final
3243 values for the job. The job's jmJobStateReasons1 object
3244 *SHOULD* contain one of: completedSuccessfully,
3245 completedWithWarnings, or completedWithErrors values.
3246
3247 This is a type 2 enumeration. See Section 3.7.1.2."
3248 SYNTAX INTEGER {
3249 unknown(2),
3250 pending(3),
3251 pendingHeld(4),
3252 processing(5),
3253 processingStopped(6),
3254 canceled(7),
3255 aborted(8),
3256 completed(9)
3257 }


```
3258
3259
3260 JmAttributeTypeTC ::= TEXTUAL-CONVENTION
3261     STATUS      current
3262     DESCRIPTION
3263         "The type of the attribute which identifies the attribute.
3264
3265     NOTE - The enum assignments are grouped logically with values
3266     assigned in groups of 20, so that additional values may be
3267     registered in the future and assigned a value that is part of
3268     their logical grouping.
3269
3270     Values in the range 2**30 to 2**31-1 are reserved for private
3271     or experimental usage. This range corresponds to the same
3272     range reserved in IPP. Implementers are warned that use of
3273     such values may conflict with other implementations.
3274     Implementers are encouraged to request registration of enum
3275     values following the procedures in Section 3.7.1.
3276
3277     See Section 3.2 entitled 'The Attribute Mechanism' for a
3278     description of this textual-convention and its use in the
3279     jmAttributeTable. See Section 3.3.8 for the specification of
3280     each attribute. The comment(s) after each enum assignment
3281     specifies the data type(s) of the attribute.
3282
3283     This is a type 2 enumeration. See Section 3.7.1.2."
3284
3285     SYNTAX      INTEGER {
3286         other(1),                -- Integer32 (-2..2147483647)
3287                                 -- AND/OR
3288                                 -- OCTET STRING(SIZE(0..63))
3289
3290         -- Job State attributes:
3291         jobStateReasons2(3),     -- JmJobStateReasons2TC
3292         jobStateReasons3(4),     -- JmJobStateReasons3TC
3293         jobStateReasons4(5),     -- JmJobStateReasons4TC
3294         processingMessage(6),    -- JmUTF8StringTC (SIZE(0..63))
3295         processingMessageNaturalLangTag(7),
3296                                 -- OCTET STRING(SIZE(0..63))
3297         jobCodedCharSet(8),      -- CodedCharSet
3298         jobNaturalLanguageTag(9), -- OCTET STRING(SIZE(0..63))
3299
```

```

3300      -- Job Identification attributes:
3301      jobURI(20),                -- OCTET STRING(SIZE(0..63))
3302      jobAccountName(21),        -- OCTET STRING(SIZE(0..63))
3303      serverAssignedJobName(22), -- JmJobStringTC (SIZE(0..63))
3304      jobName(23),               -- JmJobStringTC (SIZE(0..63))
3305      jobServiceTypes(24),       -- JmJobServiceTypesTC
3306      jobSourceChannelIndex(25), -- Integer32 (0..2147483647)
3307      jobSourcePlatformType(26), -- JmJobSourcePlatformTypeTC
3308      submittingServerName(27),  -- JmJobStringTC (SIZE(0..63))
3309      submittingApplicationName(28), -- JmJobStringTC (SIZE(0..63))
3310      jobOriginatingHost(29),   -- JmJobStringTC (SIZE(0..63))
3311      deviceNameRequested(30),  -- JmJobStringTC (SIZE(0..63))
3312      queueNameRequested(31),   -- JmJobStringTC (SIZE(0..63))
3313      physicalDevice(32),       -- hrDeviceIndex
3314                                 -- AND/OR
3315                                 -- JmUTF8StringTC (SIZE(0..63))
3316      numberOfDocuments(33),    -- Integer32 (-2..2147483647)
3317      fileName(34),             -- JmJobStringTC (SIZE(0..63))
3318      documentName(35),         -- JmJobStringTC (SIZE(0..63))
3319      jobComment(36),           -- JmJobStringTC (SIZE(0..63))
3320      documentFormatIndex(37),  -- Integer32 (0..2147483647)
3321      documentFormat(38),       -- PrtInterpreterLangFamilyTC
3322                                 -- AND/OR
3323                                 -- OCTET STRING(SIZE(0..63))
3324
3325      -- Job Parameter attributes:
3326      jobPriority(50),            -- Integer32 (-2..100)
3327      jobProcessAfterDateAndTime(51), -- DateAndTime (SNMPv2-TC)
3328      jobHold(52),              -- JmBooleanTC
3329      jobHoldUntil(53),         -- JmJobStringTC (SIZE(0..63))
3330      outputBin(54),            -- Integer32 (0..2147483647)
3331                                 -- AND/OR
3332                                 -- JmJobStringTC (SIZE(0..63))
3333      sides(55),                -- Integer32 (-2..2)
3334      finishing(56),            -- JmFinishingTC
3335
3336      -- Image Quality attributes:
3337      printQualityRequested(70),  -- JmPrintQualityTC
3338      printQualityUsed(71),       -- JmPrintQualityTC
3339      printerResolutionRequested(72), -- JmPrinterResolutionTC
3340      printerResolutionUsed(73),  -- JmPrinterResolutionTC
3341      tonerEcomonyRequested(74),  -- JmTonerEconomyTC
3342      tonerEcomonyUsed(75),       -- JmTonerEconomyTC
3343      tonerDensityRequested(76),  -- Integer32 (-2..100)
3344      tonerDensityUsed(77),       -- Integer32 (-2..100)
3345

```

```

3346      -- Job Progress attributes:
3347      jobCopiesRequested(90),          -- Integer32 (-2..2147483647)
3348      jobCopiesCompleted(91),         -- Integer32 (-2..2147483647)
3349      documentCopiesRequested(92),    -- Integer32 (-2..2147483647)
3350      documentCopiesCompleted(93),    -- Integer32 (-2..2147483647)
3351      jobKOctetsTransferred(94),      -- Integer32 (-2..2147483647)
3352      sheetCompletedCopyNumber(95),   -- Integer32 (-2..2147483647)
3353      sheetCompletedDocumentNumber(96),
3354                                          -- Integer32 (-2..2147483647)
3355      jobCollationType(97),           -- JmJobCollationTypeTC
3356
3357      -- Impression attributes:
3358      impressionsSpooled(110),         -- Integer32 (-2..2147483647)
3359      impressionsSentToDevice(111),   -- Integer32 (-2..2147483647)
3360      impressionsInterpreted(112),    -- Integer32 (-2..2147483647)
3361      impressionsCompletedCurrentCopy(113),
3362                                          -- Integer32 (-2..2147483647)
3363      fullColorImpressionsCompleted(114),
3364                                          -- Integer32 (-2..2147483647)
3365      highlightColorImpressionsCompleted(115),
3366                                          -- Integer32 (-2..2147483647)
3367
3368      -- Page attributes:
3369      pagesRequested(130),             -- Integer32 (-2..2147483647)
3370      pagesCompleted(131),            -- Integer32 (-2..2147483647)
3371      pagesCompletedCurrentCopy(132), -- Integer32 (-2..2147483647)
3372
3373      -- Sheet attributes:
3374      sheetsRequested(150),            -- Integer32 (-2..2147483647)
3375      sheetsCompleted(151),           -- Integer32 (-2..2147483647)
3376      sheetsCompletedCurrentCopy(152), -- Integer32 (-2..2147483647)
3377
3378      -- Resource attributes:
3379      mediumRequested(170),             -- JmMediumTypeTC
3380                                          -- AND/OR
3381                                          -- JmJobStringTC (SIZE(0..63))
3382      mediumConsumed(171),             -- Integer32 (-2..2147483647)
3383                                          -- AND
3384                                          -- JmJobStringTC (SIZE(0..63))
3385      colorantRequested(172),          -- Integer32 (-2..2147483647)
3386                                          -- AND/OR
3387                                          -- JmJobStringTC (SIZE(0..63))
3388      colorantConsumed(173),           -- Integer32 (-2..2147483647)
3389                                          -- AND/OR
3390                                          -- JmJobStringTC (SIZE(0..63))
3391      mediumTypeConsumed(174),          -- Integer32 (-2..2147483647)
3392                                          -- AND
3393                                          -- JmJobStringTC (SIZE(0..63))
3394      mediumSizeConsumed(175),         -- Integer32 (-2..2147483647)
3395                                          -- AND
3396                                          -- JmJobStringTC (SIZE(0..63))
3397

```

```
3398      -- Time attributes:
3399      jobSubmissionToServerTime(190), -- JmTimeStampTC
3400                                         -- AND/OR
3401                                         -- DateAndTime
3402      jobSubmissionTime(191),          -- JmTimeStampTC
3403                                         -- AND/OR
3404                                         -- DateAndTime
3405      jobStartedBeingHeldTime(192),   -- JmTimeStampTC
3406                                         -- AND/OR
3407                                         -- DateAndTime
3408      jobStartedProcessingTime(193),  -- JmTimeStampTC
3409                                         -- AND/OR
3410                                         -- DateAndTime
3411      jobCompletionTime(194),        -- JmTimeStampTC
3412                                         -- AND/OR
3413                                         -- DateAndTime
3414      jobProcessingCPUTime(195)      -- Integer32 (-2..2147483647)
3415  }
```



```

3463         faxIn                                0x10
3464             The job contains some instructions that specify receive fax
3465
3466         faxOut                                0x20
3467             The job contains some instructions that specify sending fax
3468
3469         getFile                                0x40
3470             The job contains some instructions that specify accessing
3471             files or documents
3472
3473         putFile                                0x80
3474             The job contains some instructions that specify storing
3475             files or documents
3476
3477         mailList                               0x100
3478             The job contains some instructions that specify
3479             distribution of documents using an electronic mail system.
3480
3481             These bit definitions are the equivalent of a type 2 enum
3482             except that combinations of them MAY be used together.  See
3483             section 3.7.1.2."
3484     SYNTAX      INTEGER (0..2147483647)      -- 31 bits, all but sign bit
3485
3486
3487
3488 JmJobStateReasons1TC ::= TEXTUAL-CONVENTION
3489     STATUS      current
3490     DESCRIPTION
3491         "The JmJobStateReasonsMTC (N=1..4) textual-conventions are used
3492         with the jmJobStateReasons1 object and jobStateReasonsN
3493         (N=2..4), respectively, to provide additional information
3494         regarding the current jmJobState object value.  These values
3495         MAY be used with any job state or states for which the reason
3496         makes sense.  See section 3.3.9.1 for the specification of each
3497         bit value defined for use with the JmJobStateReasons1TC.
3498
3499         These bit definitions are the equivalent of a type 2 enum
3500         except that combinations of bits may be used together.  See
3501         section 3.7.1.2."
3502     SYNTAX      INTEGER (0..2147483647)      -- 31 bits, all but sign bit
3503
3504
3505
3506 JmJobStateReasons2TC ::= TEXTUAL-CONVENTION
3507     STATUS      current
3508     DESCRIPTION
3509         "This textual-convention is used with the jobStateReasons2
3510         attribute to provides additional information regarding the
3511         jmJobState object.  See section 3.3.9.2 for the specification
3512         of JmJobStateReasons2TC.  See section 3.3.9.1 for the
3513         description under JmJobStateReasons1TC for additional
3514         information that applies to all reasons.

```

3515
3516 These bit definitions are the equivalent of a type 2 enum
3517 except that combinations of them may be used together. See
3518 section 3.7.1.2."
3519 SYNTAX INTEGER (0..2147483647) -- 31 bits, all but sign bit
3520
3521 JmJobStateReasons3TC ::= TEXTUAL-CONVENTION
3522 STATUS current
3523 DESCRIPTION
3524 "This textual-convention is used with the jobStateReasons3
3525 attribute to provides additional information regarding the
3526 jmJobState object. See section 3.3.9.3 for the specification
3527 of JmJobStateReasons3TC. See section 3.3.9.1 for the
3528 description under JmJobStateReasons1TC for additional
3529 information that applies to all reasons.
3530
3531 These bit definitions are the equivalent of a type 2 enum
3532 except that combinations of them may be used together. See
3533 section 3.7.1.2. "
3534 SYNTAX INTEGER (0..2147483647) -- 31 bits, all but sign bit
3535
3536
3537
3538
3539
3540 JmJobStateReasons4TC ::= TEXTUAL-CONVENTION
3541 STATUS current
3542 DESCRIPTION
3543 "This textual-convention is used in the jobStateReasons4
3544 attribute to provides additional information regarding the
3545 jmJobState object. See section 3.3.9.4 for the specification
3546 of JmJobStateReasons4TC. See section 3.3.9.1 for the
3547 description under JmJobStateReasons1TC for additional
3548 information that applies to all reasons.
3549
3550 These bit definitions are the equivalent of a type 2 enum
3551 except that combinations of them may be used together. See
3552 section 3.7.1.2."
3553 SYNTAX INTEGER (0..2147483647) -- 31 bits, all but sign bit

```

3554
3555
3556 jobmonMIBObjects OBJECT IDENTIFIER ::= { jobmonMIB 1 }
3557
3558 -- The General Group (MANDATORY)
3559
3560 -- The jmGeneralGroup consists entirely of the jmGeneralTable.
3561
3562 jmGeneral OBJECT IDENTIFIER ::= { jobmonMIBObjects 1 }
3563
3564 jmGeneralTable OBJECT-TYPE
3565     SYNTAX      SEQUENCE OF JmGeneralEntry
3566     MAX-ACCESS  not-accessible
3567     STATUS      current
3568     DESCRIPTION
3569         "The jmGeneralTable consists of information of a general nature
3570         that are per-job-set, but are not per-job. See Section 2
3571         entitled 'Terminology and Job Model' for the definition of a
3572         job set.
3573
3574         The MANDATORY-GROUP macro specifies that this group is
3575         MANDATORY."
3576     ::= { jmGeneral 1 }
3577
3578
3579 jmGeneralEntry OBJECT-TYPE
3580     SYNTAX      JmGeneralEntry
3581     MAX-ACCESS  not-accessible
3582     STATUS      current
3583     DESCRIPTION
3584         "Information about a job set (queue).
3585
3586         An entry SHALL exist in this table for each job set."
3587     INDEX { jmGeneralJobSetIndex }
3588     ::= { jmGeneralTable 1 }
3589
3590
3591 JmGeneralEntry ::= SEQUENCE {
3592     jmGeneralJobSetIndex      Integer32 (1..32767),
3593     jmGeneralNumberOfActiveJobs Integer32 (0..2147483647),
3594     jmGeneralOldestActiveJobIndex Integer32 (0..2147483647),
3595     jmGeneralNewestActiveJobIndex Integer32 (0..2147483647),
3596     jmGeneralJobPersistence   Integer32 (15..2147483647),
3597     jmGeneralAttributePersistence Integer32 (15..2147483647),
3598     jmGeneralJobSetName      JmUTF8StringTC (SIZE(0..63))
3599 }

```



```
3600
3601 jmGeneralJobSetIndex OBJECT-TYPE
3602     SYNTAX      Integer32 (1..32767)
3603     MAX-ACCESS  not-accessible
3604     STATUS      current
3605     DESCRIPTION
3606         "A unique value for each job set in this MIB.  The jmJobTable
3607         and jmAttributeTable tables have this same index as their
3608         primary index.
3609
3610         The value(s) of the jmGeneralJobSetIndex SHALL be persistent
3611         across power cycles, so that clients that have retained
3612         jmGeneralJobSetIndex values will access the same job sets upon
3613         subsequent power-up.
3614
3615         An implementation that has only one job set, such as a printer
3616         with a single queue, SHALL hard code this object with the value
3617         1.
3618
3619         See Section 2 entitled 'Terminology and Job Model' for the
3620         definition of a job set.
3621         Corresponds to the first index in jmJobTable and
3622         jmAttributeTable."
3623     ::= { jmGeneralEntry 1 }
3624
3625
3626 jmGeneralNumberOfActiveJobs OBJECT-TYPE
3627     SYNTAX      Integer32 (0..2147483647)
3628     MAX-ACCESS  read-only
3629     STATUS      current
3630     DESCRIPTION
3631         "The current number of 'active' jobs in the jmJobIDTable,
3632         jmJobTable, and jmAttributeTable, i.e., the total number of
3633         jobs that are in the pending, processing, or processingStopped
3634         states.  See the JmJobStateTC textual-convention for the exact
3635         specification of the semantics of the job states."
3636     DEFVAL      { 0 }      -- no jobs
3637     ::= { jmGeneralEntry 2 }
```

```
3638
3639 jmGeneralOldestActiveJobIndex OBJECT-TYPE
3640     SYNTAX      Integer32 (0..2147483647)
3641     MAX-ACCESS  read-only
3642     STATUS      current
3643     DESCRIPTION
3644         "The jmJobIndex of the oldest job that is still in one of the
3645         'active' states (pending, processing, or processingStopped).
3646         In other words, the index of the 'active' job that has been in
3647         the job tables the longest.
3648
3649         If there are no active jobs, the agent SHALL set the value of
3650         this object to 0.
3651
3652         See Section 3.2 entitled 'The Job Tables and the Oldest Active
3653         and Newest Active Indexes' for a description of the usage of
3654         this object."
3655     DEFVAL      { 0 }          -- no active jobs
3656     ::= { jmGeneralEntry 3 }
3657
3658
3659
3660 jmGeneralNewestActiveJobIndex OBJECT-TYPE
3661     SYNTAX      Integer32 (0..2147483647)
3662     MAX-ACCESS  read-only
3663     STATUS      current
3664     DESCRIPTION
3665         "The jmJobIndex of the newest job that is in one of the
3666         'active' states (pending, processing, or processingStopped).
3667         In other words, the index of the 'active' job that has been
3668         most recently added to the job tables.
3669
3670         When all jobs become 'inactive', i.e., enter the pendingHeld,
3671         completed, canceled, or aborted states, the agent SHALL set the
3672         value of this object to 0.
3673
3674         See Section 3.2 entitled 'The Job Tables and the Oldest Active
3675         and Newest Active Indexes' for a description of the usage of
3676         this object."
3677     DEFVAL      { 0 }          -- no active jobs
3678     ::= { jmGeneralEntry 4 }
```

```
3679
3680 jmGeneralJobPersistence OBJECT-TYPE
3681     SYNTAX      Integer32 (15..2147483647)
3682     UNITS       "seconds"
3683     MAX-ACCESS  read-only
3684     STATUS      current
3685     DESCRIPTION
3686         "The minimum time in seconds for this instance of the Job Set
3687         that an entry SHALL remain in the jmJobIDTable and jmJobTable
3688         after processing has completed, i.e., the minimum time in
3689         seconds starting when the job enters the completed, canceled,
3690         or aborted state.
3691
3692         Configuring this object is implementation-dependent.
3693
3694         This value SHALL be equal to or greater than the value of
3695         jmGeneralAttributePersistence. This value SHOULD be at least
3696         60 which gives a monitoring or accounting application one
3697         minute in which to poll for job data."
3698     DEFVAL      { 60 }          -- one minute
3699     ::= { jmGeneralEntry 5 }
3700
3701
3702
3703 jmGeneralAttributePersistence OBJECT-TYPE
3704     SYNTAX      Integer32 (15..2147483647)
3705     UNITS       "seconds"
3706     MAX-ACCESS  read-only
3707     STATUS      current
3708     DESCRIPTION
3709         "The minimum time in seconds for this instance of the Job Set
3710         that an entry SHALL remain in the jmAttributeTable after
3711         processing has completed , i.e., the time in seconds starting
3712         when the job enters the completed, canceled, or aborted state.
3713
3714         Configuring this object is implementation-dependent.
3715
3716         This value SHOULD be at least 60 which gives a monitoring or
3717         accounting application one minute in which to poll for job
3718         data."
3719     DEFVAL      { 60 }          -- one minute
3720     ::= { jmGeneralEntry 6 }
```

```
3721
3722 jmGeneralJobSetName OBJECT-TYPE
3723     SYNTAX      JmUTF8StringTC (SIZE(0..63))
3724     MAX-ACCESS  read-only
3725     STATUS      current
3726     DESCRIPTION
3727         "The human readable name of this job set assigned by the system
3728         administrator (by means outside of this MIB).  Typically, this
3729         name SHOULD be the name of the job queue.  If a server or
3730         device has only a single job set, this object can be the
3731         administratively assigned name of the server or device itself.
3732         This name does not need to be unique, though each job set in a
3733         single Job Monitoring MIB SHOULD have distinct names.
3734
3735         NOTE - If the job set corresponds to a single printer and the
3736         Printer MIB is implemented, this value SHOULD be the same as
3737         the prtGeneralPrinterName object in the draft Printer MIB
3738         [print-mib-draft].  If the job set corresponds to an IPP
3739         Printer, this value SHOULD be the same as the IPP 'printer-
3740         name' Printer attribute.
3741
3742         NOTE - The purpose of this object is to help the user of the
3743         job monitoring application distinguish between several job sets
3744         in implementations that support more than one job set.
3745
3746         See the OBJECT compliance macro for the minimum maximum length
3747         required for conformance."
3748     DEFVAL      { 'H }      -- empty string
3749     ::= { jmGeneralEntry 7 }
3750
3751
3752
```

```

3753
3754
3755 -- The Job ID Group (MANDATORY)
3756
3757 -- The jmJobIDGroup consists entirely of the jmJobIDTable.
3758
3759 jmJobID OBJECT IDENTIFIER ::= { jobmonMIBObjects 2 }
3760
3761 jmJobIDTable OBJECT-TYPE
3762     SYNTAX      SEQUENCE OF JmJobIDEntry
3763     MAX-ACCESS  not-accessible
3764     STATUS      current
3765     DESCRIPTION
3766         "The jmJobIDTable provides a correspondence map (1) between the
3767         job submission ID that a client uses to refer to a job and (2)
3768         the jmGeneralJobSetIndex and jmJobIndex that the Job Monitoring
3769         MIB agent assigned to the job and that are used to access the
3770         job in all of the other tables in the MIB.  If a monitoring
3771         application already knows the jmGeneralJobSetIndex and the
3772         jmJobIndex of the job it is querying, that application NEED NOT
3773         use the jmJobIDTable.
3774
3775         The MANDATORY-GROUP macro specifies that this group is
3776         MANDATORY."
3777     ::= { jmJobID 1 }
3778
3779
3780
3781 jmJobIDEntry OBJECT-TYPE
3782     SYNTAX      JmJobIDEntry
3783     MAX-ACCESS  not-accessible
3784     STATUS      current
3785     DESCRIPTION
3786         "The map from (1) the jmJobSubmissionID to (2) the
3787         jmGeneralJobSetIndex and jmJobIndex.
3788
3789         An entry SHALL exist in this table for each job currently known
3790         to the agent for all job sets and job states.  There MAY be
3791         more than one jmJobIDEntry that maps to a single job.  This
3792         many to one mapping can occur when more than one network entity
3793         along the job submission path supplies a job submission ID.
3794         See Section 3.5.  However, each job SHALL appear once and in
3795         one and only one job set."
3796     INDEX { jmJobSubmissionID }
3797     ::= { jmJobIDTable 1 }
3798
3799 JmJobIDEntry ::= SEQUENCE {
3800     jmJobSubmissionID      OCTET STRING(SIZE(48)),
3801     jmJobIDJobSetIndex     Integer32 (0..32767),
3802     jmJobIDJobIndex        Integer32 (0..2147483647)
3803 }

```

```
3804
3805 jmJobSubmissionID OBJECT-TYPE
3806     SYNTAX      OCTET STRING(SIZE(48))
3807     MAX-ACCESS  not-accessible
3808     STATUS      current
3809     DESCRIPTION
3810         "A quasi-unique 48-octet fixed-length string ID which
3811         identifies the job within a particular client-server
3812         environment.  There are multiple formats for the
3813         jmJobSubmissionID.  Each format SHALL be uniquely identified.
3814         See the JmJobSubmissionIDTypeTC textual convention.  Each
3815         format SHALL be registered using the procedures of a type 2
3816         enum.  See section 3.7.3 entitled: 'PWG Registration of Job
3817         Submission Id Formats'.
3818
3819         If the requester (client or server) does not supply a job
3820         submission ID in the job submission protocol, then the
3821         recipient (server or device) SHALL assign a job submission ID
3822         using any of the standard formats that have been reserved for
3823         agents and adding the final 8 octets to distinguish the ID from
3824         others submitted from the same requester.
3825
3826         The monitoring application, whether in the client or running
3827         separately, MAY use the job submission ID to help identify
3828         which jmJobIndex was assigned by the agent, i.e., in which row
3829         the job information is in the other tables.
3830
3831         NOTE - fixed-length is used so that a management application
3832         can use a shortened GetNext varbind (in SNMPv1 and SNMPv2) in
3833         order to get the next submission ID, disregarding the remainder
3834         of the ID in order to access jobs independent of the trailing
3835         identifier part, e.g., to get all jobs submitted by a
3836         particular jmJobOwner or submitted from a particular MAC
3837         address.
3838
3839         See the JmJobSubmissionIDTypeTC textual convention.
3840         See APPENDIX B - Support of Job Submission Protocols."
3841 ::= { jmJobIDEntry 1 }
```

```
3842
3843 jmJobIDJobSetIndex OBJECT-TYPE
3844     SYNTAX      Integer32 (0..32767)
3845     MAX-ACCESS  read-only
3846     STATUS      current
3847     DESCRIPTION
3848         "This object contains the value of the jmGeneralJobSetIndex for
3849         the job with the jmJobSubmissionID value, i.e., the job set
3850         index of the job set in which the job was placed when that
3851         server or device accepted the job. This 16-bit value in
3852         combination with the jmJobIDJobIndex value permits the
3853         management application to access the other tables to obtain the
3854         job-specific objects for this job.
3855
3856         See jmGeneralJobSetIndex in the jmGeneralTable."
3857     DEFVAL      { 0 }      -- 0 indicates no job set index
3858     ::= { jmJobIDEntry 2 }
3859
3860
3861
3862 jmJobIDJobIndex OBJECT-TYPE
3863     SYNTAX      Integer32 (0..2147483647)
3864     MAX-ACCESS  read-only
3865     STATUS      current
3866     DESCRIPTION
3867         "This object contains the value of the jmJobIndex for the job
3868         with the jmJobSubmissionID value, i.e., the job index for the
3869         job when the server or device accepted the job. This value, in
3870         combination with the jmJobIDJobSetIndex value, permits the
3871         management application to access the other tables to obtain the
3872         job-specific objects for this job.
3873
3874         See jmJobIndex in the jmJobTable."
3875     DEFVAL      { 0 }      -- 0 indicates no jmJobIndex value.
3876     ::= { jmJobIDEntry 3 }
3877
3878
```

```

3879
3880
3881 -- The Job Group (MANDATORY)
3882
3883 -- The jmJobGroup consists entirely of the jmJobTable.
3884
3885 jmJob OBJECT IDENTIFIER ::= { jobmonMIBObjects 3 }
3886
3887 jmJobTable OBJECT-TYPE
3888     SYNTAX      SEQUENCE OF JmJobEntry
3889     MAX-ACCESS  not-accessible
3890     STATUS      current
3891     DESCRIPTION
3892         "The jmJobTable consists of basic job state and status
3893         information for each job in a job set that (1) monitoring
3894         applications need to be able to access in a single SNMP Get
3895         operation, (2) that have a single value per job, and (3) that
3896         SHALL always be implemented.
3897
3898         The MANDATORY-GROUP macro specifies that this group is
3899         MANDATORY."
3900     ::= { jmJob 1 }
3901
3902
3903
3904 jmJobEntry OBJECT-TYPE
3905     SYNTAX      JmJobEntry
3906     MAX-ACCESS  not-accessible
3907     STATUS      current
3908     DESCRIPTION
3909         "Basic per-job state and status information.
3910
3911         An entry SHALL exist in this table for each job, no matter what
3912         the state of the job is. Each job SHALL appear in one and only
3913         one job set.
3914
3915         See Section 3.2 entitled 'The Job Tables'."
3916     INDEX { jmGeneralJobSetIndex, jmJobIndex }
3917     ::= { jmJobTable 1 }
3918
3919 JmJobEntry ::= SEQUENCE {
3920     jmJobIndex          Integer32 (1..2147483647),
3921     jmJobState          JmJobStateTC,
3922     jmJobStateReasons1 JmJobStateReasons1TC,
3923     jmNumberOfInterveningJobs Integer32 (-2..2147483647),
3924     jmJobKOctetsPerCopyRequested Integer32 (-2..2147483647),
3925     jmJobKOctetsProcessed Integer32 (-2..2147483647),
3926     jmJobImpressionsPerCopyRequested Integer32 (-2..2147483647),
3927     jmJobImpressionsCompleted Integer32 (-2..2147483647),
3928     jmJobOwner          JmJobStringTC (SIZE(0..63))
3929 }

```



```
3930
3931 jmJobIndex OBJECT-TYPE
3932     SYNTAX      Integer32 (1..2147483647)
3933     MAX-ACCESS  not-accessible
3934     STATUS      current
3935     DESCRIPTION
3936         "The sequential, monotonically increasing identifier index for
3937         the job generated by the server or device when that server or
3938         device accepted the job. This index value permits the
3939         management application to access the other tables to obtain the
3940         job-specific row entries.
3941
3942         See Section 3.2 entitled 'The Job Tables and the Oldest Active
3943         and Newest Active Indexes'.
3944         See Section 3.5 entitled 'Job Identification'.
3945         See also jmGeneralNewestActiveJobIndex for the largest value of
3946         jmJobIndex.
3947         See JmJobSubmissionIDTypeTC for a limit on the size of this
3948         index if the agent represents it as an 8-digit decimal number."
3949     ::= { jmJobEntry 1 }
3950
3951
3952
3953 jmJobState OBJECT-TYPE
3954     SYNTAX      JmJobStateTC
3955     MAX-ACCESS  read-only
3956     STATUS      current
3957     DESCRIPTION
3958         "The current state of the job (pending, processing, completed,
3959         etc.). Agents SHALL implement only those states which are
3960         appropriate for the particular implementation. However,
3961         management applications SHALL be prepared to receive all the
3962         standard job states.
3963
3964         The final value for this object SHALL be one of: completed,
3965         canceled, or aborted. The minimum length of time that the
3966         agent SHALL maintain MIB data for a job in the completed,
3967         canceled, or aborted state before removing the job data from
3968         the jmJobIDTable and jmJobTable is specified by the value of
3969         the jmGeneralJobPersistence object."
3970     DEFVAL      { unknown }      -- default is unknown
3971     ::= { jmJobEntry 2 }
```

```
3972
3973 jmJobStateReasons1 OBJECT-TYPE
3974     SYNTAX      JmJobStateReasons1TC
3975     MAX-ACCESS  read-only
3976     STATUS      current
3977     DESCRIPTION
3978         "Additional information about the job's current state, i.e.,
3979         information that augments the value of the job's jmJobState
3980         object.
3981
3982         Implementation of any reason values is OPTIONAL, but an agent
3983         SHOULD return any reason information available. These values
3984         MAY be used with any job state or states for which the reason
3985         makes sense. Since the Job State Reasons will be more dynamic
3986         than the Job State, it is recommended that a job monitoring
3987         application read this object every time jmJobState is read.
3988         When the agent cannot provide a reason for the current state of
3989         the job, the value of the jmJobStateReasons1 object and
3990         jobStateReasonsN attributes SHALL be 0.
3991
3992         The jobStateReasonsN (N=2..4) attributes provide further
3993         additional information about the job's current state."
3994     DEFVAL      { 0 }          -- no reasons
3995     ::= { jmJobEntry 3 }
3996
3997
3998
3999 jmNumberOfInterveningJobs OBJECT-TYPE
4000     SYNTAX      Integer32 (-2..2147483647)
4001     MAX-ACCESS  read-only
4002     STATUS      current
4003     DESCRIPTION
4004         "The number of jobs that are expected to complete processing
4005         before this job has completed processing according to the
4006         implementation's queuing algorithm, if no other jobs were to be
4007         submitted. In other words, this value is the job's queue
4008         position. The agent SHALL return a value of 0 for this
4009         attribute when the job is the next job to complete processing
4010         (or has completed processing)."
4011     DEFVAL      { 0 }          -- default is no intervening jobs.
4012     ::= { jmJobEntry 4 }
```

```
4013
4014 jmJobKOctetsPerCopyRequested OBJECT-TYPE
4015     SYNTAX      Integer32 (-2..2147483647)
4016     MAX-ACCESS  read-only
4017     STATUS      current
4018     DESCRIPTION
4019         "The total size in K (1024) octets of the document(s) being
4020         requested to be processed in the job.  The agent SHALL round
4021         the actual number of octets up to the next highest K.  Thus 0
4022         octets is represented as '0', 1-1024 octets is represented as
4023         '1', 1025-2048 is represented as '2', etc.
4024
4025         In computing this value, the server/device SHALL NOT include
4026         the multiplicative factors contributed by (1) the number of
4027         document copies, and (2) the number of job copies, independent
4028         of whether the device can process multiple copies of the job or
4029         document without making multiple passes over the job or
4030         document data and independent of whether the output is collated
4031         or not.  Thus the server/device computation is independent of
4032         the implementation and indicates the size of the document(s)
4033         measured in K octets independent of the number of copies."
4034     DEFVAL      { -2 }      -- the default is unknown(-2)
4035     ::= { jmJobEntry 5 }
4036
4037
4038
4039 jmJobKOctetsProcessed OBJECT-TYPE
4040     SYNTAX      Integer32 (-2..2147483647)
4041     MAX-ACCESS  read-only
4042     STATUS      current
4043     DESCRIPTION
4044         "The total number of octets processed by the server or device
4045         measured in units of K (1024) octets so far.  The agent SHALL
4046         round the actual number of octets processed up to the next
4047         higher K.  Thus 0 octets is represented as '0', 1-1024 octets
4048         is represented as '1', 1025-2048 octets is '2', etc.  For
4049         printing devices, this value is the number interpreted by the
4050         page description language interpreter rather than what has been
4051         marked on media.
4052
4053         For implementations where multiple copies are produced by the
4054         interpreter with only a single pass over the data, the final
4055         value SHALL be equal to the value of the
4056         jmJobKOctetsPerCopyRequested object.  For implementations where
4057         multiple copies are produced by the interpreter by processing
4058         the data for each copy, the final value SHALL be a multiple of
4059         the value of the jmJobKOctetsPerCopyRequested object.
4060
4061         NOTE - See the impressionsCompletedCurrentCopy and
4062         pagesCompletedCurrentCopy attributes for attributes that are
4063         reset on each document copy.
4064
```

4065 NOTE - The jmJobKOctetsProcessed object can be used with the
4066 jmJobKOctetsPerCopyRequested object to provide an indication of
4067 the relative progress of the job, provided that the
4068 multiplicative factor is taken into account for some
4069 implementations of multiple copies."
4070 DEFVAL { 0 } -- default is no octets processed.
4071 ::= { jmJobEntry 6 }
4072
4073
4074 jmJobImpressionsPerCopyRequested OBJECT-TYPE
4075 SYNTAX Integer32 (-2..2147483647)
4076 MAX-ACCESS read-only
4077 STATUS current
4078 DESCRIPTION
4079 "The total size in number of impressions of the document(s)
4080 submitted.
4081
4082 In computing this value, the server/device SHALL NOT include
4083 the multiplicative factors contributed by (1) the number of
4084 document copies, and (2) the number of job copies, independent
4085 of whether the device can process multiple copies of the job or
4086 document without making multiple passes over the job or
4087 document data and independent of whether the output is collated
4088 or not. Thus the server/device computation is independent of
4089 the implementation and reflects the size of the document(s)
4090 measured in impressions independent of the number of copies.
4091
4092 See the definition of the term 'impression' in Section 2."
4093 DEFVAL { -2 } -- default is unknown(-2)
4094 ::= { jmJobEntry 7 }
4095
4096
4097 jmJobImpressionsCompleted OBJECT-TYPE
4098 SYNTAX Integer32 (-2..2147483647)
4099 MAX-ACCESS read-only
4100 STATUS current
4101 DESCRIPTION
4102 "The total number of impressions completed for this job so far.
4103 For printing devices, the impressions completed includes
4104 interpreting, marking, and stacking the output. For other
4105 types of job services, the number of impressions completed
4106 includes the number of impressions processed.
4107
4108 NOTE - See the impressionsCompletedCurrentCopy and
4109 pagesCompletedCurrentCopy attributes for attributes that are
4110 reset on each document copy.
4111
4112 NOTE - The jmJobImpressionsCompleted object can be used with
4113 the jmJobImpressionsPerCopyRequested object to provide an
4114 indication of the relative progress of the job, provided that
4115 the multiplicative factor is taken into account for some
4116 implementations of multiple copies.

```
4117
4118     See the definition of the term 'impression' in Section 2 and
4119     the counting example in Section 3.4 entitled 'Monitoring Job
4120     Progress'."
4121     DEFVAL      { 0 }      -- default is no octets
4122     ::= { jmJobEntry 8 }
4123
4124
4125
4126 jmJobOwner OBJECT-TYPE
4127     SYNTAX      JmJobStringTC (SIZE(0..63))
4128     MAX-ACCESS  read-only
4129     STATUS      current
4130     DESCRIPTION
4131         "The coded character set name of the user that submitted the
4132         job.  The method of assigning this user name will be system
4133         and/or site specific but the method MUST ensure that the name
4134         is unique to the network that is visible to the client and
4135         target device.
4136
4137         This value SHOULD be the most authenticated name of the user
4138         submitting the job.
4139
4140         See the OBJECT compliance macro for the minimum maximum length
4141         required for conformance."
4142     DEFVAL      { ''H }      -- default is empty string
4143     ::= { jmJobEntry 9 }
4144
4145
```

```
4146
4147
4148 -- The Attribute Group (MANDATORY)
4149
4150 -- The jmAttributeGroup consists entirely of the jmAttributeTable.
4151 --
4152 -- Implementation of the objects in this group is MANDATORY.
4153 -- See Section 3.1 entitled 'Conformance Considerations'.
4154 -- An agent SHALL implement any attribute if (1) the server or device
4155 -- supports the functionality represented by the attribute and (2) the
4156 -- information is available to the agent.
4157
4158 jmAttribute OBJECT IDENTIFIER ::= { jobmonMIBObjects 4 }
4159
4160
4161
4162 jmAttributeTable OBJECT-TYPE
4163     SYNTAX          SEQUENCE OF JmAttributeEntry
4164     MAX-ACCESS     not-accessible
4165     STATUS          current
4166     DESCRIPTION
4167         "The jmAttributeTable SHALL contain attributes of the job and
4168         document(s) for each job in a job set.  Instead of allocating
4169         distinct objects for each attribute, each attribute is
4170         represented as a separate row in the jmAttributeTable.
4171
4172         The MANDATORY-GROUP macro specifies that this group is
4173         MANDATORY.  An agent SHALL implement any attribute if (1) the
4174         server or device supports the functionality represented by the
4175         attribute and (2) the information is available to the agent. "
4176     ::= { jmAttribute 1 }
4177
4178
4179
```

```

4180 jmAttributeEntry OBJECT-TYPE
4181     SYNTAX          JmAttributeEntry
4182     MAX-ACCESS     not-accessible
4183     STATUS         current
4184     DESCRIPTION
4185         "Attributes representing information about the job and
4186         document(s) or resources required and/or consumed.
4187
4188         Each entry in the jmAttributeTable is a per-job entry with an
4189         extra index for each type of attribute (jmAttributeTypeIndex)
4190         that a job can have and an additional index
4191         (jmAttributeInstanceIndex) for those attributes that can have
4192         multiple instances per job.  The jmAttributeTypeIndex object
4193         SHALL contain an enum type that indicates the type of attribute
4194         (see the JmAttributeTypeTC textual-convention).  The value of
4195         the attribute SHALL be represented in either the
4196         jmAttributeValueAsInteger or jmAttributeValueAsOctets objects,
4197         and/or both, as specified in the JmAttributeTypeTC textual-
4198         convention.
4199
4200         The agent SHALL create rows in the jmAttributeTable as the
4201         server or device is able to discover the attributes either from
4202         the job submission protocol itself or from the document PDL.
4203         As the documents are interpreted, the interpreter MAY discover
4204         additional attributes and so the agent adds additional rows to
4205         this table.  As the attributes that represent resources are
4206         actually consumed, the usage counter contained in the
4207         jmAttributeValueAsInteger object is incremented according to
4208         the units indicated in the description of the JmAttributeTypeTC
4209         enum.
4210
4211         The agent SHALL maintain each row in the jmAttributeTable for
4212         at least the minimum time after a job completes as specified by
4213         the jmGeneralAttributePersistence object.
4214
4215         Zero or more entries SHALL exist in this table for each job in
4216         a job set.
4217
4218         See Section 3.3 entitled 'The Attribute Mechanism' for a
4219         description of the jmAttributeTable."
4220     INDEX { jmGeneralJobSetIndex, jmJobIndex, jmAttributeTypeIndex,
4221            jmAttributeInstanceIndex }
4222     ::= { jmAttributeTable 1 }
4223
4224 JmAttributeEntry ::= SEQUENCE {
4225     jmAttributeTypeIndex          JmAttributeTypeTC,
4226     jmAttributeInstanceIndex      Integer32 (1..32767),
4227     jmAttributeValueAsInteger     Integer32 (-2..2147483647),
4228     jmAttributeValueAsOctets     OCTET STRING(SIZE(0..63))
4229 }

```

```
4230
4231 jmAttributeTypeIndex OBJECT-TYPE
4232     SYNTAX      JmAttributeTypeTC
4233     MAX-ACCESS  not-accessible
4234     STATUS      current
4235     DESCRIPTION
4236         "The type of attribute that this row entry represents.
4237
4238         The type MAY identify information about the job or document(s)
4239         or MAY identify a resource required to process the job before
4240         the job start processing and/or consumed by the job as the job
4241         is processed.
4242
4243         Examples of job attributes (i.e., apply to the job as a whole)
4244         that have only one instance per job include:
4245         jobCopiesRequested(90), documentCopiesRequested(92),
4246         jobCopiesCompleted(91), documentCopiesCompleted(93), while
4247         examples of job attributes that may have more than one instance
4248         per job include: documentFormatIndex(37), and
4249         documentFormat(38).
4250
4251         Examples of document attributes (one instance per document)
4252         include: fileName(34), and documentName(35).
4253
4254         Examples of required and consumed resource attributes include:
4255         pagesRequested(130), mediumRequested(170), pagesCompleted(131),
4256         and mediumConsumed(171), respectively."
4257 ::= { jmAttributeEntry 1 }
4258
4259
4260
4261 jmAttributeInstanceIndex OBJECT-TYPE
4262     SYNTAX      Integer32 (1..32767)
4263     MAX-ACCESS  not-accessible
4264     STATUS      current
4265     DESCRIPTION
4266         "A running 16-bit index of the attributes of the same type for
4267         each job.  For those attributes with only a single instance per
4268         job, this index value SHALL be 1.  For those attributes that
4269         are a single value per document, the index value SHALL be the
4270         document number, starting with 1 for the first document in the
4271         job.  Jobs with only a single document SHALL use the index
4272         value of 1.  For those attributes that can have multiple values
4273         per job or per document, such as documentFormatIndex(37) or
4274         documentFormat(38), the index SHALL be a running index for the
4275         job as a whole, starting at 1."
4276 ::= { jmAttributeEntry 2 }
```


4277
4278 jmAttributeValueAsInteger OBJECT-TYPE
4279 SYNTAX Integer32 (-2..2147483647)
4280 MAX-ACCESS read-only
4281 STATUS current
4282 DESCRIPTION
4283 "The integer value of the attribute. The value of the
4284 attribute SHALL be represented as an integer if the enum
4285 description in the JmAttributeTypeTC textual-convention
4286 definition has the tag: 'INTEGER:'.
4287
4288 Depending on the enum definition, this object value MAY be an
4289 integer, a counter, an index, or an enum, depending on the
4290 jmAttributeTypeIndex value. The units of this value are
4291 specified in the enum description.
4292
4293 For those attributes that are accumulating job consumption as
4294 the job is processed as specified in the JmAttributeTypeTC
4295 textual-convention, SHALL contain the final value after the job
4296 completes processing, i.e., this value SHALL indicate the total
4297 usage of this resource made by the job.
4298
4299 A monitoring application is able to copy this value to a
4300 suitable longer term storage for later processing as part of an
4301 accounting system.
4302
4303 Since the agent MAY add attributes representing resources to
4304 this table while the job is waiting to be processed or being
4305 processed, which can be a long time before any of the resources
4306 are actually used, the agent SHALL set the value of the
4307 jmAttributeValueAsInteger object to 0 for resources that the
4308 job has not yet consumed.
4309
4310 Attributes for which the concept of an integer value is
4311 meaningless, such as fileName(34), jobName, and
4312 processingMessage, do not have the 'INTEGER:' tag in the
4313 JmAttributeTypeTC definition and so an agent SHALL always
4314 return a value of '-1' to indicate 'other' for the value of the
4315 jmAttributeValueAsInteger object for these attributes.
4316
4317 For attributes which do have the 'INTEGER:' tag in the
4318 JmAttributeTypeTC definition, if the integer value is not (yet)
4319 known, the agent either (1) SHALL not materialize the row in
4320 the jmAttributeTable until the value is known or (2) SHALL
4321 return a '-2' to represent an 'unknown' counting integer value,
4322 a '0' to represent an 'unknown' index value, and a '2' to
4323 represent an 'unknown(2)' enum value."
4324 DEFVAL { -2 } -- default value is unknown(-2)
4325 ::= { jmAttributeEntry 3 }

```
4326
4327 jmAttributeValueAsOctets OBJECT-TYPE
4328     SYNTAX      OCTET STRING(SIZE(0..63))
4329     MAX-ACCESS  read-only
4330     STATUS      current
4331     DESCRIPTION
4332         "The octet string value of the attribute.  The value of the
4333         attribute SHALL be represented as an OCTET STRING if the enum
4334         description in the JmAttributeTypeTC textual-convention
4335         definition has the tag: 'OCTETS:'."
4336
4337         Depending on the enum definition, this object value MAY be a
4338         coded character set string (text), such as 'JmUTF8StringTC', or
4339         a binary octet string, such as 'DateAndTime'.
4340
4341         Attributes for which the concept of an octet string value is
4342         meaningless, such as pagesCompleted, do not have the tag
4343         'OCTETS:' in the JmAttributeTypeTC definition and so the agent
4344         SHALL always return a zero length string for the value of the
4345         jmAttributeValueAsOctets object.
4346
4347         For attributes which do have the 'OCTETS:' tag in the
4348         JmAttributeTypeTC definition, if the OCTET STRING value is not
4349         (yet) known, the agent either SHALL NOT materialize the row in
4350         the jmAttributeTable until the value is known or SHALL return a
4351         zero-length string."
4352     DEFVAL      { ''H } -- empty string
4353     ::= { jmAttributeEntry 4 }
```

```
4354
4355
4356 -- The Mirror Attribute Group (OPTIONAL)
4357
4358 -- The jmMirrorAttrGroup consists entirely of the jmMirrorAttrTable.
4359 --
4360 -- Implementation of the objects in this group is OPTIONAL.
4361 -- See Section 3.1 entitled 'Conformance Considerations'.
4362 -- The jmMirrorAttrTable complements the MANDATORY jmAttributeTable.
4363 --
4364 -- The jmMirrorAttrTable provides access to all of the attributes that
4365 -- an implementation supports, sorted by attribute type (traditional
4366 -- SNMP MIB access), rather than being sorted by job set and job index
4367 -- (modern object-oriented access) as in the analogous
4368 -- jmAttributeTable.
4369
4370 jmMirrorAttr      OBJECT IDENTIFIER ::= { jobmonMIBObjects 5 }
4371
4372 jmMirrorAttrTable OBJECT-TYPE
4373     SYNTAX          SEQUENCE OF JmMirrorAttrEntry
4374     MAX-ACCESS      not-accessible
4375     STATUS          current
4376     DESCRIPTION
4377         "The jmMirrorAttrTable is an OPTIONAL table which provides
4378         identical attributes to the jmAttributeTable but with a
4379         different index structure.  See jmAttributeTable for further
4380         details.
4381
4382         See Section 3.3 entitled 'The Attribute Mechanism' for a
4383         description of the jmMirrorAttrTable."
4384     ::= { jmMirrorAttr 1 }
4385
4386
4387
```

```

4388 jmMirrorAttrEntry OBJECT-TYPE
4389     SYNTAX      JmMirrorAttrEntry
4390     MAX-ACCESS  not-accessible
4391     STATUS      current
4392     DESCRIPTION
4393         "The attributes that represent information about each job and
4394         documents or resources required and/or consumed.
4395
4396         Each entry in jmMirrorAttrTable is a per-attribute entry with a
4397         primary index for each type of attribute (jmMirrorAttrTypeIndex)
4398         that a job can have and secondary indices which specify job set
4399         (jmJobSetIndex), job instance (jmJobIndex), and attribute
4400         instance (jmMirrorAttrInstanceIndex).
4401
4402         An agent which implements the jmMirrorAttrTable SHALL create
4403         and maintain a row in the jmMirrorAttrTable for each
4404         corresponding row in the jmAttributeTable."
4405     INDEX { jmMirrorAttrTypeIndex, jmGeneralJobSetIndex, jmJobIndex,
4406            jmMirrorAttrInstanceIndex }
4407     ::= { jmMirrorAttrTable 1 }
4408
4409 JmMirrorAttrEntry ::= SEQUENCE {
4410     jmMirrorAttrTypeIndex      JmAttributeTypeTC,
4411     jmMirrorAttrInstanceIndex  Integer32 (1..32767),
4412     jmMirrorAttrValueAsInteger Integer32 (-2..2147483647),
4413     jmMirrorAttrValueAsOctets  OCTET STRING(SIZE(0..63))
4414 }
4415
4416 jmMirrorAttrTypeIndex OBJECT-TYPE
4417     SYNTAX      JmAttributeTypeTC
4418     MAX-ACCESS  not-accessible
4419     STATUS      current
4420     DESCRIPTION
4421         "The type of attribute that this row entry represents.
4422
4423         See jmAttributeTypeIndex in jmAttributeTable for complete
4424         description."
4425     ::= { jmMirrorAttrEntry 1 }
4426
4427 jmMirrorAttrInstanceIndex OBJECT-TYPE
4428     SYNTAX      Integer32 (1..32767)
4429     MAX-ACCESS  not-accessible
4430     STATUS      current
4431     DESCRIPTION
4432         "The instance of attribute that this row entry represents.
4433
4434         See jmAttributeInstanceIndex in jmAttributeTable for complete
4435         description."
4436     ::= { jmMirrorAttrEntry 2 }
4437

```

```
4438
4439 jmMirrorAttrValueAsInteger OBJECT-TYPE
4440     SYNTAX      Integer32 (-2..2147483647)
4441     MAX-ACCESS  read-only
4442     STATUS      current
4443     DESCRIPTION
4444         "The integer value of the attribute.
4445
4446         See jmAttributeValueAsInteger in jmAttributeTable for complete
4447         description."
4448     DEFVAL      { -2 }          -- default value is unknown(-2)
4449     ::= { jmMirrorAttrEntry 3 }
4450
4451 jmMirrorAttrValueAsOctets OBJECT-TYPE
4452     SYNTAX      OCTET STRING(SIZE(0..63))
4453     MAX-ACCESS  read-only
4454     STATUS      current
4455     DESCRIPTION
4456         "The octet string value of the attribute.
4457
4458         See jmAttributeValueAsOctets in jmAttributeTable for complete
4459         description."
4460     DEFVAL      { ''H }        -- empty string
4461     ::= { jmMirrorAttrEntry 4 }
```

```
4462
4463
4464 -- The System Group (MANDATORY)
4465 -- (This group was added in version 1.3 of this MIB).
4466
4467 -- The jmMirrorAttrGroup consists entirely of objects that summarize
4468 -- the implementation of this MIB on a system.
4469
4470 jmSystem          OBJECT IDENTIFIER ::= { jobmonMIBObjects 6 }
4471
4472 jmSystemVersionString OBJECT-TYPE
4473     SYNTAX          JmUTF8StringTC
4474     MAX-ACCESS      read-only
4475     STATUS          current
4476     DESCRIPTION
4477         "The minor and minor version of this MIB implemented by this
4478         system.
4479
4480         The format of the string SHALL be the ASCII major version
4481         number followed by an ASCII PERIOD (.), followed by the ASCII
4482         minor version number, i.e., '1.3' for this version."
4483     DEFVAL          { '312E33'H }          -- version 1.3
4484     ::= { jmSystem 1 }
4485
4486 jmSystemOptionSupport OBJECT-TYPE
4487     SYNTAX          INTEGER(0..2147483647)  -- biggest int 2**31 - 1
4488     MAX-ACCESS      read-only
4489     STATUS          current
4490     DESCRIPTION
4491         "The options of the MIB specification that this implementation
4492         supports specified as a bit mask.
4493
4494         The current set of values (which may be extended in the future)
4495         is given below:
4496
4497         1 : jmMirrorAttrGroup          -- 2**0    OPTIONAL
4498
4499         Example: An implementation supporting the jmMirrorAttrGroup
4500         would return an integer value of { 1 }.
4501
4502         This object helps a management application determine which MIB
4503         options are supported in this system."
4504     DEFVAL          { 0 }          -- no options are required
4505     ::= { jmSystem 2 }
4506
```

```
4507
4508 jmSystemAttrIntegerSupport OBJECT-TYPE
4509     SYNTAX      OCTET STRING (SIZE (0..63))
4510     MAX-ACCESS  read-only
4511     STATUS      current
4512     DESCRIPTION
4513         "A bit array indicating which attributes of the MIB this
4514         implementation supports with meaningful integer values.
4515
4516         The value of this object is a sparse bit array in which bit n
4517         is a 1 if attribute n is supported with the
4518         jmAttributeValueAsInteger object with meaningful values, where
4519         n is the value of the enumerated attribute type in the
4520         JmAttributeTypeTC used in jmAttributeTypeIndex (and the
4521         jmMirrorAttrTypeIndex if the jmMirrorAttrTable is implemented).
4522         Bit n MUST be 0 (or beyond the end of the returned bit array),
4523         if attribute n is not supported or is always returned with a '-
4524         1'(other) or '-2'(unknown) value.
4525
4526         The high order bit of the first octet in this octet string
4527         corresponds to an attribute type of 0 (reserved), i.e., the bit
4528         string uses the Big Endian numbering convention. Compare with
4529         the BITS data type in SMIV2 [SMIV2-SMI] which has the same
4530         format but requires contiguous enumerated bits. Trailing
4531         octets in the octet string that contain only zero bits MUST NOT
4532         be returned.
4533
4534         Note: private attributes cannot be represented in this bit
4535         array because their enum values are in the range 2**30 to
4536         2**31-1. See section 3.3.8.
4537
4538         Example: An implementation supporting the attributes:
4539         jobStateReasons2(3), jobStateReasons3(4), and jobName(23)
4540         would return a one-octet string value of { '18'H }, since
4541         jobName is an octet string value, not an integer value.
4542
4543         This object helps a management application determine which
4544         attributes with meaningful integer values MAY be present on
4545         jobs in this system."
4546     DEFVAL      { 'H } -- no attributes are required
4547     ::= { jmSystem 3 }
4548
```

```
4549
4550 jmSystemAttrOctetsSupport OBJECT-TYPE
4551     SYNTAX      OCTET STRING (SIZE (0..63))
4552     MAX-ACCESS  read-only
4553     STATUS      current
4554     DESCRIPTION
4555         "A bit array indicating which attributes of the MIB this
4556         implementation supports with meaningful octet string values.
4557
4558         The format and semantics of this object is the same as
4559         jmSystemAttrIntegerSupport, except that bit n indicates that
4560         attribute n supports the jmAttributeValueAsOctets object with
4561         meaningful values, instead of the jmAttributeValueAsInteger
4562         object.  Bit n MUST be 0 (or beyond the end of the returned bit
4563         array), if attribute n is not supported or is always returned
4564         as a zero-length octet string value.
4565
4566         If an implementation supports both jmAttributeValueAsInteger
4567         and jmAttributeValueAsOctets with meaningful values for
4568         attribute n, bit n MUST appear in both bit array objects with a
4569         1 value.
4570
4571         Example:  An implementation supporting the attributes:
4572         jobStateReasons2(3), jobStateReasons3(4), and jobName(23)
4573         would return a three-octet string value of { '000001'H }, since
4574         jobStateReasons2 and jobStateReasons3 are integer values, not
4575         octet string values.
4576
4577         This object helps a management application determine which
4578         attributes with meaningful octet string values MAY be present
4579         on jobs in this system."
4580     DEFVAL      { 'H }          -- no attributes are required
4581     ::= { jmSystem 4 }
4582
```



```

4583
4584 jmSystemAttrMultiRowSupport OBJECT-TYPE
4585     SYNTAX      OCTET STRING (SIZE (0..63))
4586     MAX-ACCESS  read-only
4587     STATUS      current
4588     DESCRIPTION
4589         "A bit array indicating which MULTI-ROW attributes of the MIB
4590         this implementation supports with multiple integer values
4591         and/or multiple octet string values.
4592
4593         The format of this object is the same as the
4594         jmSystemAttrIntegerSupport and jmSystemAttrOctetsSupport
4595         objects.  Bit n MUST be 1, if attribute n is actually supported
4596         with more than one integer and/or more than one octet string
4597         value.  Bit n MUST be 0 (or beyond the end of the returned bit
4598         array), if attribute n is not supported, is always returned as
4599         a single integer value, or as a single octet string value.  For
4600         every bit n that is a 1 in this bit array, there MUST be a
4601         corresponding 1 for bit n in either jmSystemAttrIntegerSupport,
4602         jmSystemAttrOctetsSupport, or both.
4603
4604         Example:  Consider an implementation supporting:
4605         (a) the jobStateReasons2(3), jobStateReasons3(4) SINGLE-ROW
4606         integer attributes
4607         (b) the jobName(23) SINGLE-ROW octet string attribute
4608         (c) more than one integer value for the mediumRequested(170)
4609         and mediumConsumed(171) MULTI-ROW attributes AND
4610         (d) more than one octet string value for the fileName(34),
4611         documentName(35), and mediumConsumed(171) MULTI-ROW attributes
4612         (e) no octet string values for mediumRequested(170).
4613         Such an implementation would return:
4614         jmSystemAttrIntegerSupport 22 octets:
4615             { '18000000 00000000 00000000 00000000 00000000 0030'H }
4616         jmSystemAttrOctetsSupport 22 octets:
4617             { '00000100 30000000 00000000 00000000 00000000 0010'H }
4618         jmSystemAttrMultiRowSupport 22 octets:
4619             { '00000000 30000000 00000000 00000000 00000000 0030'H }
4620
4621         Example:  Consider an implementation that supports the
4622         fileName(34) MULTI-ROW attribute, but does not support more
4623         than one document per job.  Such an implementation would NOT
4624         return a 1 bit for bit 34 in jmSystemAttrMultiRowSupport, since
4625         such an implementation would never return more than one
4626         fileName value for a job.  It would return a zero-length
4627         string, since it never returns more than one value.
4628
4629         This object helps a management application determine which
4630         attributes may return more than one integer value or more than
4631         one octet string value on jobs in this system."
4632     DEFVAL      { 'H } -- no attributes are required
4633     ::= { jmSystem 5 }

```

```
4634 -- Notifications and Trapping
4635 -- Reserved for the future
4636
4637 jobmonMIBNotifications OBJECT IDENTIFIER ::= { jobmonMIB 2 }
4638
4639
4640
4641 -- Conformance Information
4642
4643 jmMIBConformance OBJECT IDENTIFIER ::= { jobmonMIB 3 }
4644
4645
4646
4647 -- compliance statements
4648 jmMIBCompliance MODULE-COMPLIANCE
4649     STATUS current
4650     DESCRIPTION
4651         "The compliance statement for agents that implement the
4652         job monitoring MIB."
4653     MODULE -- this module
4654     MANDATORY-GROUPS {
4655         jmGeneralGroup, jmJobIDGroup, jmJobGroup, jmAttributeGroup,
4656         jmSystemGroup }
4657
4658     GROUP jmMirrorAttrGroup
4659     DESCRIPTION
4660         "The mirror attribute group (sorted by attribute type).
4661         Implementation of this group is OPTIONAL.
4662
4663         An agent that implements the jmMirrorAttrTable SHALL create and
4664         maintain for the same time a row in the jmMirrorAttrTable for
4665         each corresponding row in the jmAttributeTable."
4666
4667     OBJECT jmGeneralJobSetName
4668     SYNTAX JmUTF8StringTC (SIZE(0..8))
4669     DESCRIPTION
4670         "Only 8 octets maximum string length NEED be supported by the
4671         agent."
4672
4673     OBJECT jmJobOwner
4674     SYNTAX JmJobStringTC (SIZE(0..16))
4675     DESCRIPTION
4676         "Only 16 octets maximum string length NEED be supported by the
4677         agent."
4678
4679 -- There are no CONDITIONALLY MANDATORY groups.
4680
4681 ::= { jmMIBConformance 1 }
4682
```

```
4683 jmMIBGroups      OBJECT IDENTIFIER ::= { jmMIBConformance 2 }
4684
4685 jmGeneralGroup OBJECT-GROUP
4686     OBJECTS {
4687         jmGeneralNumberOfActiveJobs,    jmGeneralOldestActiveJobIndex,
4688         jmGeneralNewestActiveJobIndex,  jmGeneralJobPersistence,
4689         jmGeneralAttributePersistence,  jmGeneralJobSetName}
4690     STATUS current
4691     DESCRIPTION
4692         "The general group."
4693     ::= { jmMIBGroups 1 }
4694
4695
4696
4697 jmJobIDGroup OBJECT-GROUP
4698     OBJECTS {
4699         jmJobIDJobSetIndex, jmJobIDJobIndex }
4700     STATUS current
4701     DESCRIPTION
4702         "The job ID group."
4703     ::= { jmMIBGroups 2 }
4704
4705
4706
4707 jmJobGroup OBJECT-GROUP
4708     OBJECTS {
4709         jmJobState, jmJobStateReasons1, jmNumberOfInterveningJobs,
4710         jmJobKOctetsPerCopyRequested, jmJobKOctetsProcessed,
4711         jmJobImpressionsPerCopyRequested, jmJobImpressionsCompleted,
4712         jmJobOwner }
4713     STATUS current
4714     DESCRIPTION
4715         "The job group."
4716     ::= { jmMIBGroups 3 }
4717
4718
4719
4720 jmAttributeGroup OBJECT-GROUP
4721     OBJECTS {
4722         jmAttributeValueAsInteger, jmAttributeValueAsOctets }
4723     STATUS current
4724     DESCRIPTION
4725         "The attribute group."
4726     ::= { jmMIBGroups 4 }
4727
4728
```

```
4729 jmMirrorAttrGroup OBJECT-GROUP
4730     OBJECTS {
4731         jmMirrorAttrValueAsInteger, jmMirrorAttrValueAsOctets }
4732     STATUS current
4733     DESCRIPTION
4734         "The mirror attribute group (sorted by attribute type).
4735         Implementation of this group is OPTIONAL.
4736
4737         An agent which implements the jmMirrorAttrTable SHALL create
4738         and maintain for the same time a row in the jmMirrorAttrTable
4739         for each corresponding row in the jmAttributeTable."
4740     ::= { jmMIBGroups 5 }
4741
4742
4743 jmSystemGroup OBJECT-GROUP
4744     OBJECTS {
4745         jmSystemVersionString, jmSystemOptionSupport,
4746         jmSystemAttrIntegerSupport,
4747         jmSystemAttrOctetsSupport,
4748         jmSystemAttrMultiRowSupport }
4749     STATUS current
4750     DESCRIPTION
4751         "The system group."
4752     ::= { jmMIBGroups 6 }
4753
4754
4755 END
```

4756

4757 5 Appendix A - Implementing the Job Life Cycle

4758 The job object has well-defined states and client operations that
4759 affect the transition between the job states. Internal server and
4760 device actions also affect the transitions of the job between the job
4761 states. These states and transitions are referred to as the job's *life*
4762 *cycle*.

4763 Not all implementations of job submission protocols have all of the
4764 states of the job model specified here. The job model specified here
4765 is intended to be a superset of most implementations. It is the
4766 purpose of the agent to map the particular implementation's job life
4767 cycle onto the one specified here. The agent MAY omit any states not
4768 implemented. Only the processing and completed states are required to
4769 be implemented by an agent. However, a conforming management
4770 application SHALL be prepared to accept any of the states in the job
4771 life cycle specified here, so that the management application can
4772 interoperate with any conforming agent.

4773 The job states are intended to be user visible. The agent SHALL make
4774 these states visible in the MIB, but only for the subset of job states
4775 that the implementation has. Some implementations MAY need to have
4776 sub-states of these user-visible states. The jmJobStateReasons1 object
4777 and the jobStateReasonsN (N=2..4) attributes can be used to represent
4778 the sub-states of the jobs.

4779 Job states are intended to last a user-visible length of time in most
4780 implementations. However, some jobs may pass through some states in
4781 zero time in some situations and/or in some implementations.

4782 The job model does not specify how accounting and auditing is
4783 implemented, except to assume that accounting and auditing logs are
4784 separate from the job life cycle and last longer than job entries in
4785 the MIB. Jobs in the completed, aborted, or canceled states are not
4786 logs, since jobs in these states are accessible via SNMP protocol
4787 operations and SHALL be removed from the Job Monitoring MIB tables
4788 after a site-settable or implementation-defined period of time. An
4789 accounting application MAY copy accounting information incrementally to
4790 an accounting log as a job processes, or MAY be copied while the job is
4791 in the canceled, aborted, or completed states, depending on
4792 implementation. The same is true for auditing logs.

4793 The jmJobState object specifies the standard job states. The normal
4794 job state transitions are shown in the state transition diagram
4795 presented in Table 1.

4796

4797 6 APPENDIX B - Support of Job Submission Protocols

4798 A companion PWG document, entitled "Job Submission Protocol Mapping
4799 Recommendations for the Job Monitoring MIB" [protomap] contains the
4800 recommended usage of each of the objects and attributes in this MIB
4801 with a number of job submission protocols. In particular, which job
4802 submission ID format should be used is indicated for each job
4803 submission protocol.

4804 Some job submission protocols have support for the client to specify a
4805 job submission ID. A second approach is to enhance the document format
4806 to embed the job submission ID in the document data. This second
4807 approach is independent of the job submission protocol. This appendix
4808 lists some examples of these approaches.

4809 Some PJL implementations wrap a banner page as a PJL job around a job
4810 submitted by a client. If this results in multiple job submission IDs,
4811 the agent SHALL create multiple jmJobIDEntry rows in the jmJobIDTable
4812 that each point to the same job entry in the job tables. See the
4813 specification of the jmJobIDEntry.

4814 7 References

4815 [BCP-11] [Bradner S.](#), [Hovey R.](#), "The Organizations Involved in the IETF
4816 Standards Process", 1996/10/29 (RFC 2028)

4817 [GB2312] GB 2312-1980, "Chinese People's Republic of China (PRC) mixed
4818 one byte and two byte coded character set"

4819 [hr-mib] P. Grillo, S. Waldbusser, "Host Resources MIB", RFC 1514,
4820 September 1993

4821 [iana] J. Reynolds, and J. Postel, "Assigned Numbers", STD 2, RFC 1700,
4822 ISI, October 1994.

4823 [IANA-charsets] Coded Character Sets registered by IANA and assigned an
4824 enum value for use in the CodedCharSet textual convention imported from
4825 the Printer MIB. See [ftp://ftp.isi.edu/in-](ftp://ftp.isi.edu/in-notes/iana/assignments/character-sets)
4826 [notes/iana/assignments/character-sets](ftp://ftp.isi.edu/in-notes/iana/assignments/character-sets)

4827 [iana-media-types] IANA Registration of MIME media types (MIME content
4828 types/subtypes). See <ftp://ftp.isi.edu/in-notes/iana/assignments/>

4829 [ipp-model] Internet Printing Protocol/1.0: Model and Semantics, work
4830 in progress on the IETF standards track. See [draft-ietf-ipp-model-](draft-ietf-ipp-model-09.txt)
4831 [09.txt](http://www.pwg.org/ipp/index.html). See also <http://www.pwg.org/ipp/index.html>

- 4832 [ISO-639] ISO 639:1988 (E/F) - Code for Representation of names of
4833 languages - The International Organization for Standardization, 1st
4834 edition, 1988.
- 4835 [ISO-646] ISO/IEC 646:1991, "Information technology -- ISO 7-bit coded
4836 character set for information interchange", JTC1/SC2.
- 4837 [ISO-2022] ISO/IEC 2022:1994 - "Information technology -- Character
4838 code structure and extension techniques", JTC1/SC2.
- 4839 [ISO-3166] ISO 3166:1988 (E/F) - Codes for representation of names of
4840 countries - The International Organization for Standardization, 3rd
4841 edition, 1988-08-15."
- 4842 [ISO-8859-1] ISO/IEC 8859-1:1987, "Information technology -- 8-bit
4843 single byte coded graphic character sets - Part 1: Latin alphabet No.
4844 1, JTC1/SC2."
- 4845 [ISO-10646] ISO/IEC 10646-1:1993, "Information technology -- Universal
4846 Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and
4847 Basic Multilingual Plane, JTC1/SC2.
- 4848 [iso-dpa] ISO/IEC 10175-1:1996 "Information technology -- Text and
4849 Office Systems -- Document Printing Application (DPA) -- Part 1:
4850 Abstract service definition and procedures. See
4851 <ftp://ftp.pwg.org/pub/pwg/dpa/>
- 4852 [JIS X0208] JIS X0208-1990, "Japanese two byte coded character set."
- 4853 [mib-II] MIB-II, RFC 1213.
- 4854 [print-mib] Smith, R., Wright, F., Hastings, T., Zilles, S. and
4855 Gyllenskog, J., "Printer MIB", RFC 1759, proposed IETF standard, March
4856 1995. See also [print-mib-draft].
- 4857 [print-mib-draft] Turner, R., "Printer MIB", work in progress, on the
4858 standards track as a draft standard: <draft-ietf-printmib-mib-info-
4859 02.txt>, January 22, 1999.
- 4860 [protomap] Bergman, R., "Job Submission Protocol Mapping
4861 Recommendations for the Job Monitoring MIB," work in progress as an
4862 informational RFC. See <draft-bergman-printmib-job-protomap-03.txt>,
4863 February 10, 1998.
- 4864 [pwg] The Printer Working Group is a printer industry consortium open
4865 to any individuals. For more information, access the PWG web page:
4866 <http://www.pwg.org>
- 4867 [RFC1738] Berners-Lee, T., Masinter, L., McCahill, M., "Uniform
4868 Resource Locators (URL)", RFC 1738, December 1994.

- 4869 [RFC1766] Avelstrand, H., "Tags for the Identification of Languages",
4870 RFC 1766, March 1995.
- 4871 [RFC2026] S. Bradner, "The Internet Standards Process -- Revision 3",
4872 RFC 2026, October 1996.
- 4873 [RFC2119] S. Bradner, "Keywords for use in RFCs to Indicate Requirement
4874 Levels", RFC 2119, March 1997.
- 4875 [RFC2277] H. Alvestrand, "IETF Policy on Character Sets and
4876 Languages" RFC 2277, January 1998.
- 4877 [RFC2278] N. Freed, J. Postel: "IANA CharSet Registration
4878 Procedures", RFC 2278, January 1998.
- 4879 [SMIv2-SMI] J. Case, et al. "Structure of Management Information for
4880 Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC
4881 1902, January 1996.
- 4882 [SMIv2-TC] J. Case, et al. "Textual Conventions for Version 2 of the
4883 Simple Network Management Protocol (SNMPv2)", RFC 1903, January 1996.
- 4884 [tipsi] IEEE 1284.1, Transport-independent Printer System Interface
4885 (TIPSI).
- 4886 [URI-spec] Berners-Lee, T., Fielding, R., Masinter, L., "Uniform
4887 Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.
- 4888 [US-ASCII] Coded Character Set - 7-bit American Standard Code for
4889 Information Interchange, ANSI X3.4-1986.
- 4890 [UTF-8] F. Yergeau, "UTF-8, a transformation format of ISO 10646", RFC
4891 2279, January 1998.

4892 8 Notices

4893 The IETF takes no position regarding the validity or scope of any
4894 intellectual property or other rights that might be claimed to pertain
4895 to the implementation or use of the technology described in this
4896 document or the extent to which any license under such rights might or
4897 might not be available; neither does it represent that it has made any
4898 effort to identify any such rights. Information on the IETF's
4899 procedures with respect to rights in standards-track and standards-
4900 related documentation can be found in BCP-11[BCP-11]. Copies of claims
4901 of rights made available for publication and any assurances of licenses
4902 to be made available, or the result of an attempt made to obtain a
4903 general license or permission for the use of such proprietary rights by
4904 implementers or users of this specification can be obtained from the
4905 IETF Secretariat.

4906 The IETF invites any interested party to bring to its attention any
4907 copyrights, patents or patent applications, or other proprietary rights
4908 which may cover technology that may be required to practice this
4909 standard. Please address the information to the IETF Executive
4910 Director.

4911 Copyright (C) The Internet Society (1999). All Rights Reserved.

4912 This document and translations of it may be copied and furnished to
4913 others, and derivative works that comment on or otherwise explain it or
4914 assist in its implementation may be prepared, copied, published and
4915 distributed, in whole or in part, without restriction of any kind,
4916 provided that the above copyright notice and this paragraph are
4917 included on all such copies and derivative works. However, this
4918 document itself may not be modified in any way, such as by removing the
4919 copyright notice or references to the Internet Society or other
4920 Internet organizations, except as needed for the purpose of developing
4921 Internet standards in which case the procedures for copyrights defined
4922 in the Internet Standards process must be followed, or as required to
4923 translate it into languages other than English.

4924 The limited permissions granted above are perpetual and will not be
4925 revoked by the Internet Society or its successors or assigns.

4926 This document and the information contained herein is provided on an
4927 "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING
4928 TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
4929 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL
4930 NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR
4931 FITNESS FOR A PARTICULAR PURPOSE.

4932 9 Author's Addresses
4933 Ron Bergman
4934 Dataproducts Corp.
4935 1757 Tapo Canyon Road
4936 Simi Valley, CA 93063-3394
4937
4938 Phone: 805-578-4421
4939 Fax: 805-578-4001
4940 Email: rbergman@dpc.com
4941
4942
4943 Tom Hastings
4944 Xerox Corporation, ESAE-231
4945 737 Hawaii St.
4946 El Segundo, CA 90245
4947
4948 Phone: 310-333-6413

4949 Fax: 310-333-5514
4950 EMail: hastings@cp10.es.xerox.com
4951
4952

4953 Scott A. Isaacson
4954 Novell, Inc.
4955 122 E 1700 S
4956 Provo, UT 84606
4957
4958 Phone: 801-861-7366
4959 Fax: 801-861-4025
4960 EMail: scott_isaacson@novell.com
4961
4962

4963 Harry Lewis
4964 IBM Corporation
4965 6300 Diagonal Hwy
4966 Boulder, CO 80301
4967
4968 Phone: (303) 924-5337
4969 Fax:
4970 Email: harryl@us.ibm.com
4971
4972

4973 Send questions and comments to the Printer Working Group (PWG)
4974 using the Job Monitoring Project (JMP) Mailing List: jmp@pwg.org
4975

4976 To learn how to subscribe, send email to: jmp-request@pwg.org
4977

4978 Implementers of this specification are encouraged to join the jmp
4979 mailing list in order to participate in discussions on any
4980 clarifications needed and registration proposals for additional
4981 attributes and values being reviewed in order to achieve consensus.
4982

4983 For further information, access the PWG web page under "JMP":
4984

4985 <http://www.pwg.org/>
4986

4987 Other Participants:

4988 Chuck Adams - Tektronix
4989 Jeff Barnett - IBM
4990 Keith Carter, IBM Corporation
4991 Jeff Copeland - QMS
4992 Andy Davidson - Tektronix
4993 Roger deBry - IBM
4994 Mabry Dozier - QMS
4995 Lee Farrell - Canon
4996 Steve Gebert - IBM
4997 Robert Herriot - Sun Microsystems Inc.
4998 Shige Kanemitsu - Kyocera

4999 David Kellerman - Northlake Software
5000 Rick Landau - Digital
5001 Pete Loya - HP
5002 Ray Lutz - Cognisys
5003 Jay Martin - Underscore
5004 Mike MacKay, Novell, Inc.
5005 Stan McConnell - Xerox
5006 Carl-Uno Manros, Xerox, Corp.
5007 Pat Nogay - IBM
5008 Bob Pentecost - HP
5009 Rob Rhoads - Intel
5010 David Roach - Unisys
5011 Stuart Rowley - Kyocera
5012 Hiroyuki Sato - Canon
5013 Bob Setterbo - Adobe
5014 Gail Songer, EFI
5015 Mike Timperman - Lexmark
5016 Randy Turner - Sharp
5017 William Wagner - Digital Products
5018 Jim Walker - Dazel
5019 Chris Wellens - Interworking Labs
5020 Rob Whittle - Novell
5021 Don Wright - Lexmark
5022 Lloyd Young - Lexmark
5023 Atsushi Yuki - Kyocera
5024 Peter Zehler, Xerox, Corp.

5025 10 Change History

5026 This section summarizes the changes in each version after version 1.0
5027 in reverse chronological order.

5028 10.1 Changes to produce version 2.0, dated February 20, 1999

5029 The following changes were made to version 1.2, dated October 2, 1998
5030 to make version 2.0, dated February 20, 1999:

5031 1. Added the Mirror table.

5032 2. Moved the JmJobSubmissionIDTypeTC, JmJobStateReasons1TC,
5033 JmJobStateReasons2TC, JmJobStateReasons3TC, and JmJobStateReasons4TC
5034 assignments out of the MIB and into the Introduction.

5035 3. Added the MANDATORY jmSystemGroup that contains the
5036 jmSystemVersionString, jmSystemOptionSupport,
5037 jmSystemAttrIntegerSupport, jmSystemAttrOctetsSupport, and
5038 jmSystemAttrMultiRowSupport objects.

5039 4. Changed the version number to 2.0, since a MANDATORY table was
5040 added.

- 5041
- 5042 10.2Changes to produce version 1.2, dated October 2, 1998
- 5043 The following changes were made to version 1.1, dated October 1, 1998
5044 to make version 1.2, dated October 2, 1998:
- 5045 1. Removed all REFERENCE clauses since they referred to sections in the
5046 specification that were not in the MIB.
 - 5047 2. Moved the definitions of the attributes from the TC to a new section
5048 3.3.8 as requested by the IESG.
 - 5049 3. Removed the attributes from the Table of Contents
 - 5050 4. Added the data types as ASN.1 comments after each attribute enum.
 - 5051 5. Changed a number of occurrences of "SHALL" to "is" when they were
5052 just definitions, rather than conformance requirements.
- 5053
- 5054 10.3Changes to produce version 1.1, dated October 1, 1998
- 5055 The following changes were made to version 1.0, dated February 3, 1998
5056 to make version 1.1, dated October 1, 1998:
- 5057 1. Clarified sections 3.3.3 and 3.3.7 so that the DEFVAL of 0 for index
5058 attributes is different from the DEFVAL for
5059 jmAttributeValueAsInteger which is -2.
 - 5060 2. Clarified the relationships of the values of the
5061 JmJobCollationTypeTC with the IPP "multiple-document-handling"
5062 attribute.
 - 5063 3. Clarified that the values of the mediumRequested(170) and
5064 mediumConsumed(171) attributes may be any of the IPP 'media' values
5065 which are media names, media size names, and input tray names.
 - 5066 4. Added the two attributes approved by the PWG for registration in
5067 April 1998: mediumTypeConsumed(174) and mediumSizeConsumed(175).
 - 5068 5. Changed "insure" to "ensure'.
 - 5069 6. Correct an incorrect reference in the jmAttributeEntry DESCRIPTION
5070 from jmJobTable to jmAttributeTable.

5071

5072 11 INDEX

5073 This index includes the textual conventions, the objects, and the
5074 attributes. Textual conventions all start with the prefix: "JM" and
5075 end with the suffix: "TC". Objects all starts with the prefix: "jm"
5076 followed by the group name. Attributes are identified with enums, and
5077 so start with any lower case letter and have no special prefix.

5078

5079 colorantConsumed, 43
5080 colorantRequested, 42
5081 deviceNameRequested, 32
5082 documentCopiesCompleted, 37
5083 documentCopiesRequested, 37
5084 documentFormat, 34
5085 documentFormatIndex, 33
5086 documentName, 33
5087 fileName, 33
5088 finishing, 36
5089 fullColorImpressionsCompleted, 39
5090 highlightColorImpressionsCompleted, 40
5091 impressionsCompletedCurrentCopy, 39
5092 impressionsInterpreted, 39
5093 impressionsSentToDevice, 39
5094 impressionsSpooled, 39
5095 jmAttributeInstanceIndex, 104
5096 jmAttributeTypeIndex, 104
5097 JmAttributeTypeTC, 81
5098 jmAttributeValueAsInteger, 105
5099 jmAttributeValueAsOctets, 106
5100 JmBooleanTC, 75
5101 JmFinishingTC, 73
5102 jmGeneralAttributePersistence, 91
5103 jmGeneralJobPersistence, 91
5104 jmGeneralJobSetIndex, 89
5105 jmGeneralJobSetName, 92
5106 jmGeneralNewestActiveJobIndex, 90
5107 jmGeneralNumberOfActiveJobs, 89
5108 jmGeneralOldestActiveJobIndex, 90
5109 JmJobCollationTypeTC, 77
5110 jmJobIDJobIndex, 95
5111 jmJobIDJobSetIndex, 95
5112 jmJobImpressionsCompleted, 100
5113 jmJobImpressionsPerCopyRequested, 100
5114 jmJobIndex, 97
5115 jmJobKOctetsPerCopyRequested, 99
5116 jmJobKOctetsProcessed, 99
5117 jmJobOwner, 101
5118 JmJobServiceTypesTC, 85

5119 JmJobSourcePlatformTypeTC, 72
5120 jmJobState, 97
5121 jmJobStateReasons1, 98
5122 JmJobStateReasons1TC, 86
5123 JmJobStateReasons2TC, 86
5124 JmJobStateReasons3TC, 87
5125 JmJobStateReasons4TC, 87
5126 JmJobStateTC, 78
5127 JmJobStringTC, 71
5128 jmJobSubmissionID, 94
5129 JmJobSubmissionIDTypeTC, 77
5130 JmMediumTypeTC, 75
5131 jmMirrorAttrInstanceIndex, 108
5132 jmMirrorAttrTypeIndex, 108
5133 jmMirrorAttrValueAsInteger, 109
5134 jmMirrorAttrValueAsOctets, 109
5135 JmNaturalLanguageTagTC, 71
5136 jmNumberOfInterveningJobs, 98
5137 JmPrinterResolutionTC, 74
5138 JmPrintQualityTC, 74
5139 jmSystemAttrIntegerSupport, 111
5140 jmSystemAttrMultiRowSupport, 113
5141 jmSystemAttrOctetsSupport, 112
5142 jmSystemOptionSupport, 110
5143 jmSystemVersionString, 110
5144 JmTimeStampTC, 72
5145 JmTonerEconomyTC, 75
5146 JmUTF8StringTC, 71
5147 jobAccountName, 29
5148 jobCodedCharSet, 28
5149 jobCollationType, 38
5150 jobComment, 33
5151 jobCompletionTime, 45
5152 jobCopiesCompleted, 37
5153 jobCopiesRequested, 37
5154 jobHold, 35
5155 jobHoldUntil, 35
5156 jobKOctetsTransferred, 38
5157 jobName, 30
5158 jobNaturalLanguageTag, 28
5159 jobOriginatingHost, 32
5160 jobPriority, 34
5161 jobProcessAfterDateAndTime, 35
5162 jobProcessingCPUtime, 45
5163 jobServiceTypes, 31
5164 jobSourceChannelIndex, 31
5165 jobSourcePlatformType, 31
5166 jobStartedBeingHeldTime, 44
5167 jobStartedProcessingTime, 45
5168 jobStateReasons2, 26
5169 jobStateReasons3, 26
5170 jobStateReasons4, 26

5171 jobSubmissionTime, 44
5172 jobSubmissionToServerTime, 44
5173 jobURI, 29
5174 mediumConsumed, 42
5175 mediumRequested, 42
5176 mediumSizeConsumed, 43
5177 mediumTypeConsumed, 43
5178 numberOfDocuments, 32
5179 other, 26
5180 outputBin, 35
5181 pagesCompleted, 40
5182 pagesCompletedCurrentCopy, 41
5183 pagesRequested, 40
5184 physicalDevice, 32
5185 printerResolutionRequested, 36
5186 printerResolutionUsed, 36
5187 printQualityRequested, 36
5188 printQualityUsed, 36
5189 processingMessage, 27
5190 processingMessageNaturalLangTag, 27
5191 queueNameRequested, 32
5192 serverAssignedJobName, 29
5193 sheetCompletedCopyNumber, 38
5194 sheetCompletedDocumentNumber, 38
5195 sheetsCompleted, 41
5196 sheetsCompletedCurrentCopy, 41
5197 sheetsRequested, 41
5198 sides, 35
5199 submittingApplicationName, 31
5200 submittingServerName, 31
5201 tonerDensityRequested, 36
5202 tonerDensityUsed, 37
5203 tonerEcomonyRequested, 36
5204 tonerEcomonyUsed, 36

5205